MULTI-AGENT CAUSAL DISCOVERY USING LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Models (LLMs) have demonstrated significant potential in causal discovery tasks by utilizing their vast expert knowledge from extensive text corpora. However, the multi-agent capabilities of LLMs in causal discovery remain underexplored. This paper introduces a general framework to investigate this potential. The first is the Meta Agents Model, which relies exclusively on reasoning and discussions among LLM agents to conduct causal discovery. The second is the Coding Agents Model, which leverages the agents' ability to plan, write, and execute code, utilizing advanced statistical libraries for causal discovery. The third is the Hybrid Model, which integrates both the Meta Agents Model and Coding Agents Model approaches, combining the statistical analysis and reasoning skills of multiple agents. Our proposed framework shows promising results by effectively utilizing LLMs' expert knowledge, reasoning capabilities, multi-agent cooperation, and statistical causal methods. By exploring the multi-agent potential of LLMs, we aim to establish a foundation for further research in utilizing LLMs multi-agent for solving causal-related problems.

1 INTRODUCTION

Understanding causal relationships is crucial across scientific fields. While statistical causal inference is widely used, it heavily relies on assumed causal graphs. To address this limitation, data-driven methods have evolved, leading to statistical causal discovery (SCD) approaches and the creation of datasets for evaluation. Despite advancements in SCD algorithms, data-driven causal graphs without domain knowledge can be inaccurate. This inaccuracy is often due to a mismatch between SCD algorithm assumptions and real-world phenomena (Reisach et al. (2021)). Incorporating expert knowledge can mitigate this issue, but it is costly.

The advent of Large Language Models (LLMs), trained on vast amounts of data, has enabled them to acquire extensive knowledge, from common sense to specific domains such as math and science. Recent studies suggest that complex behaviors, such as writing code, generating long stories, and even reasoning capabilities, can emerge from large-scale training (Wei et al. (2023); Rozière et al. (2024); Zhao et al. (2023b); Yao et al. (2023a)). LLMs present a promising alternative for obtaining expert knowledge more accessible and affordable. Recent research (Kıcıman et al. (2023);Choi et al. (2022); Long et al. (2024); Chen et al. (2024b)) has attempted to leverage these capabilities for causal discovery based on metadata and knowledge-based reasoning, akin to human domain experts.

However, most methods have not yet fully utilized the full capacity of LLMs, i.e., LLMs' multi-agent approaches. An LLM agent can be seen as an entity with memory, reasoning, and the ability to access external tools or APIs such as a calculator, web search, and code compiler. Agent-based systems have demonstrated significant problem-solving abilities. However, a single-agent-based system sometimes still suffers from hallucinations despite having self-reflection capabilities (Li et al. (2023); Shinn et al. (2023), Madaan et al. (2023)). Inspired by the Society of Mind concept (Minsky (1988)), LLM multi-agent system discussion frameworks like MAD Liang et al. (2023), ReConcile (Chen et al. (2024a), and CMD Wang et al. (2024) have been proposed to address these issues. These LLM multi-agent systems not only achieve impressive results but also enable less capable models to perform on par with superior ones.

Despite the potential benefits, few studies have explored leveraging LLM multi-agent system capabilities in causal discovery. To address this gap, we propose a novel framework called Multi-Agent for Causality (MAC), which comprises three different models.

Method / Approach	LLMs for metadata	Statistical Approach for structured data	Agentic Ability	Multi-Agent Workflow	Introduced By
Pairwise Causal Discovery	✓	✗	✗	✗	Kıcıman et al. (2023); Zečević et al. (2023)
Various of prompt-engineering strategies	✓	✗	✗	✗	Chen et al. (2024b)
Efficiently asking question for Causal Graph Discovery Using LLMs	✓	✓	✗	✗	Jiralerspong et al. (2024)
Combining LLMs with traditional causal methods.	✓	✓	✗	✗	Vashishtha et al. (2023); Takayama et al. (2024)
Multi-Agent Causal Discover (MAC)	✓	✓	✓	✓	**Our approach**

Table 1: Comparison of Approaches for Using LLMs in Causal Discovery

The first is called **Meta Agents Model**, which includes a *Meta-Debate Module* with two debater agents and one judge agent. These agents are designed for causal discovery problems and utilize the nature of debating to find causal relationships among variables through multiple rounds of discussion. The second is called **Coding Agents Model**, which includes a *Debate-Coding Module* with four agents operating in two phases. This group leverages both the debating abilities of the agents and statistical causal algorithms. The third one is **Hybrid Model**, which hybridizes the statistical causal discovery algorithms with the reasoning skills of multiple agents to construct the causal graph.

In this research, we experiment with various LLM models across small, medium, and large scales, paired with different MAC models. We conduct an in-depth analysis of performance for each MAC model and LLM, token consumption, and identify their usage patterns and limitations. Additionally, we propose alternative solutions to address the computational costs associated with LLMs.

Our proposed framework shows promising results by effectively utilizing LLMs' expert knowledge, reasoning capabilities, multi-agent cooperation, and statistical causal methods. As far as we know, this paper is first work to explore LLMs' multi-agent abilities in a causal context. We hope that our work will lay the foundation for further research in utilizing LLM multi-agent systems for solving causal-related problems.

2 Related Works

2.1 LLMs' Agentic Workflow

A general LLM agent framework consists of core components: user request, agent/brain, planning, memory, and tools. The agent/brain acts as the main coordinator, activated by a prompt template. It can be profiled with specific details to define its role, using handcrafted, LLM-generated, or data-driven strategies. Planning employs techniques like Chain of Thought and Tree of Thoughts, and for complex tasks, feedback mechanisms like ReAct Yao et al. (2023b) and Reflexion Shinn et al. (2023) refine plans based on past actions and observations. Memory stores the agent's logs, with short-term memory for the current context and long-term memory for past behaviors. Hybrid memory combines both to enhance reasoning and experience accumulation. Tools enable interaction with external environments, such as APIs and code interpreters. Frameworks like MRKL Karpas et al. (2022), Toolformer Schick et al. (2023), Function Calling OpenAI (2024), and HuggingGPT Shen et al. (2023) integrate tools to solve tasks effectively.

However, for more complex problems where a single LLM agent may struggle, LLM-MA (multi-agent) systems excel. Current LLM-MA systems primarily employ three communication paradigms: Cooperative, Competitive, and Debating. In the Cooperative paradigm, agents collaborate towards a shared goal, typically exchanging information to enhance a collective solution Qian et al. (2023); Chen et al. (2024c). In the Competitive paradigm, agents work towards their own goals, which might conflict with those of other agents Zhao et al. (2023a). The Debating paradigm involves agents engaging in argumentative interactions, where they present and defend their viewpoints or

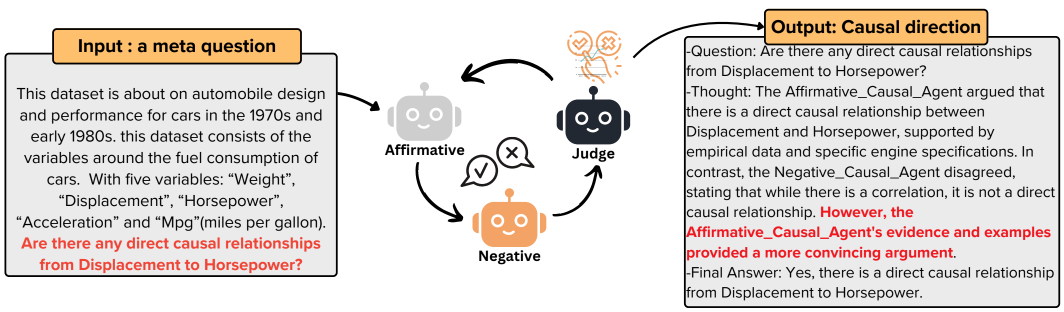


Figure 1: Meta-Debate Module

solutions while critiquing those of others. This approach is ideal for reaching a consensus or a more refined solution (Li et al. (2023); Liang et al. (2023); Xiong et al. (2023)). In this work, the debating paradigm will be implemented, as the nature of causal discovery problems requires diverse and potentially conflicting opinions to approach the truth.

2.2 Statistical and LLM-based Causal Methods

Traditional methods of statistical causal inference often depend heavily on assumed causal graphs to identify and measure causal impacts. To overcome this limitation, data-driven algorithmic approaches have been developed into statistical causal discovery (SCD) methods, encompassing both non-parametric (e.g., Spirtes et al. (2000); ; ; Yuan & Malone (2013); Huang et al. (2018); Xie et al. (2020)) and semi-parametric (e.g., Shimizu et al. (2006); Hoyer et al. (2009); Shimizu et al. (2011)) techniques. Many SCD algorithms can be systematically augmented with background knowledge and have accessible software packages. For example, the non-parametric and constraint-based Peter-Clerk (PC) algorithm (Spirtes et al. (2000)) in "causal-learn" integrates background knowledge of mandatory or forbidden directed edges. "Causal-learn" also includes the Exact Search algorithm (Shimizu et al. (2011); Yuan & Malone (2013)), a non-parametric and score-based SCD method that can incorporate background knowledge in the form of a super structure matrix of forbidden directed edges. Furthermore, the semi-parametric DirectLiNGAM (Shimizu et al. (2011)) algorithm can use prior knowledge of causal order (Inazumi et al. (2010)) in the "LiNGAM" project (Ikeuchi et al. (2023)).

In the context of knowledge-driven approaches using large language models (LLMs), applying LLMs for causal inference is relatively new. There have been a few significant efforts to use LLMs for causal inference among variables by merely prompting with the variable names, without going through the traditional SCD process with benchmark datasets (Kıcıman et al. (2023); Zečević et al. (2023)). Jiralerspong et al. (2024) even uses a breadth-first search (BFS) approach which allows it to use only a linear number of queries to have a higher efficiency. Additionally, Chen et al. (2024b) proposes 9 prompting techniques comprise ICL, 0-shot CoT (e.g. "let's think step by step"), adversarial prompt, manual CoT, and explicit function (e.g using encouraging language in prompts). Moreover, the works of Ban et al. (2023) and Vashishtha et al. (2023) introduce interesting approach by integrate LLMs into traditional data-driven approaches. However, most of above-mentioned works have not investigate the LLM-agentic work-flows for causal graph discovery **??**, which indeed requires heavy investigation on various models and graphs' scales.

3 MAC: Multi-Agent Causality Framework

3.1 Multi-Agent Causality Modules

3.1.1 Meta-Debate Module

The Meta-Debate Module is an advanced system comprising three intelligent agents: two causal debaters (an affirmative one and a negative one) and one causal judge. This structure emulates the dynamic and rigorous nature of human debate, specifically within the realm of causal discovery. The design of this debating module is meticulously crafted to ensure a thorough examination of critical elements in causal discovery, such as understanding the temporal order necessary to establish cause-and-effect relationships and identifying potential confounding variables that could distort perceived relationships between primary variables. Each side engages in active disagreement by

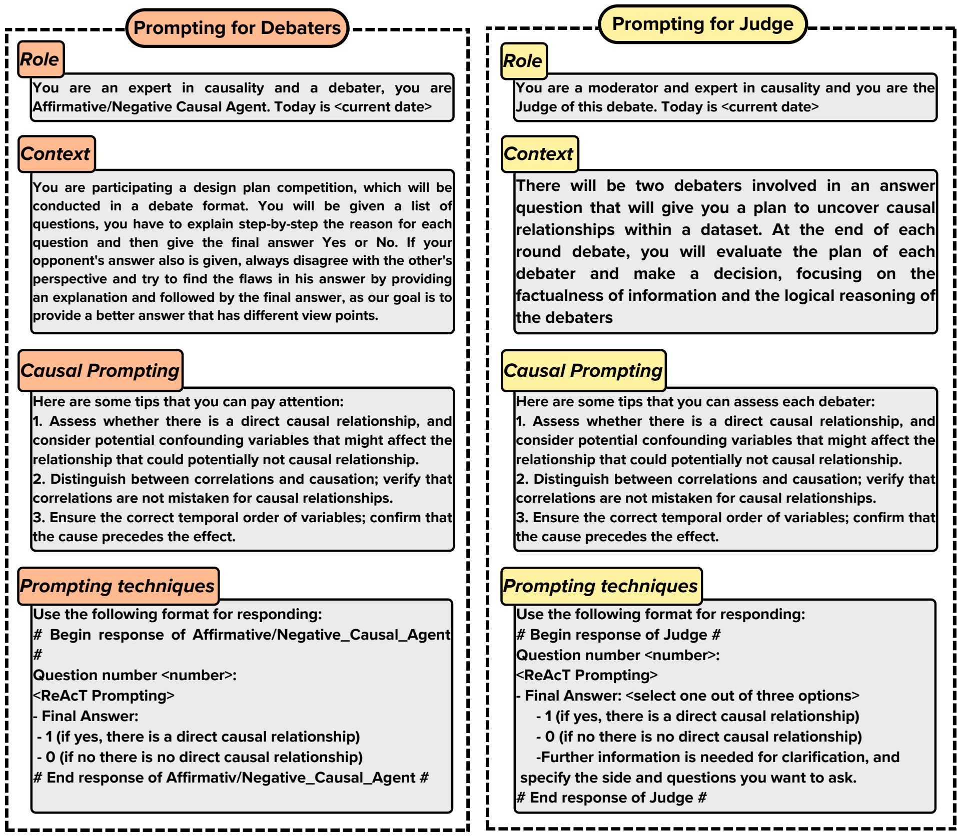


Figure 2: Prompting design of each agent in MAC, details in the Appendix A.3

presenting different opinions and viewpoints, fostering a comprehensive and robust debate. Additionally, each agent within Meta-Debate Module utilizes the ReAct prompting technique. This technique integrates the ability to dynamically formulate, modify, and refine action plans based on new information or insights gained during the debate. This integration allows the agents to engage in more sophisticated and adaptive reasoning processes, closely mimicking human-like debate and decision-making (Figure 2).

The debating process begins with a meta-question. Initially, the causal affirmative side presents its answer along with the supporting rationale. Subsequently, the causal negative side offers diverse or conflicting perspectives by providing alternative viewpoints on the same question. The causal judge then evaluates the responses from both sides, determining either a winner or identifying the need for additional clarification. If further information is required, the causal judge poses specific follow-up questions, prompting the debaters to clarify and elaborate on their initial propositions. The causal judge reaches a final verdict once all relevant information has been thoroughly examined.

For meta-questions, questions might address aspects such as the appropriate algorithm for a particular problem, the causal relationship between two variables, or a step-by-step approach for solving a causal problem, as illustrated in Figure 1. The affirmative side initiates the debating process based on the constructed input. The final output is derived from the causal judge's decision, reflecting a comprehensive evaluation of the arguments presented by both sides. This rigorous process ensures that the outcome is well-informed and considers multiple perspectives, thereby enhancing the robustness of causal discovery. A detailed visualization of this module can be found in Appendix A.2.1.

3.1.2 Debate-Coding Module

The Debate-Coding Module leverages statistical algorithms to achieve precise causal discovery through a structured two-phase process. This group consists of four agents, divided into two phases: debating the algorithm and executing the algorithm.

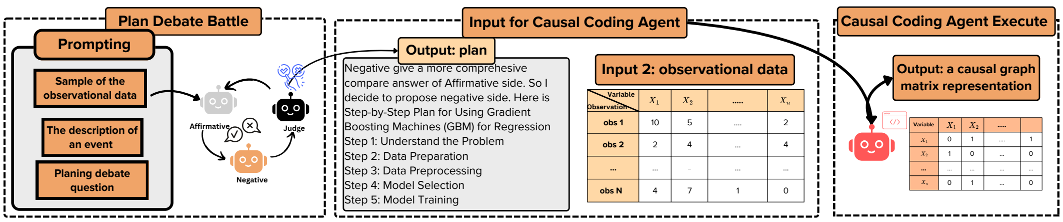


Figure 3: Debate-Coding Module

In the initial phase, three agents engage in a debate format, similar to the workflow of Meta-Debate module at Section 3.1.1. However, the difference is that the affirmative and negative agent are pre-prompts the information with 3-5 statistical causal algorithms. Additionally, the meta-question posed to these agents is specifically curated to determine which algorithm should be used given the metadata, which includes the description and structure of the data. After the debate, the output is the most suitable algorithm for the particular dataset and question. Compared to the Meta-Debate Module, an additional step in this process is that the agent participating in the debate (whether affirmative or negative) will provide a step-by-step plan for implementing the selected algorithm.

In the second phase, the causal coding executor receives the plan from the previous phase along with the observational data. The causal coding executor is responsible for writing, executing, and debugging the code. It pre-prompts the functions and provides parameters within a specific Python library[1] based on the algorithm selected by the debaters in the initial stage. This pre-prompting is crucial because LLMs can call functions from their training dataset, which may be outdated or incorrect, leading to errors and excessive debugging (details of prompting design can be found in the Appendix A.3). After executing the code, the causal coding executor outputs a matrix representation of the causal graph. A detailed visualization of this module can be found in the Appendix A.2.2.

3.2 Implementations of MAC

In this section, we will elaborate on the detailed implementation of the three models regarding their input and basic workflow.

3.2.1 Meta Agents Model

Algorithm 1 Algorithm for Meta Agents Model

Input: Data $X = [x_1, ..., x_n]$
function META_DEBATE_MODULE(query)
    Result: Response to the query about causal relationships
end function
Output: Graph G_{ij} where i and j are indices in N
for $i \leftarrow 1$ to $|X|$ **do**
    for $j \leftarrow 1$ to $i - 1$ **do**
        $G_{ij} \leftarrow$ `Meta_Debate_Module`("Are there any direct causal from $X[i]$ to $X[j]$")
    end for
    for $j \leftarrow i + 1$ to $|X|$ **do**
        $G_{ij} \leftarrow$ `Meta_Debate_Module`("Are there any direct causal relationship from $X[i]$ to $X[j]$")
    end for
end for
return Graph

The algorithm for the Meta Agents Model aims to identify direct causal relationships between variables in a dataset. It starts with the input data $X = [x_1, ..., x_n]$. The output of the algorithm is a graph $\mathcal{G}$, with its edges G_{ij}, where i and j are indices in the set of variables n

[1]More functions can be found at `https://causal-learn.readthedocs.io/en/latest/index.html`

The algorithm proceeds by iterating through each variable i from 1 to the number of variables in X. For each i, it checks all j values less than i (i.e., j ranges from 1 to $i-1$). It queries the Meta-Debate Module to check if there is a direct causal relationship from $X[i]$ to $X[j]$ and stores the result in G_{ij}. Then, it checks all j values greater than i (i.e., j ranges from $i+1$ to the number of labels in X). Those queries are meta-questions that use the Meta-Debate Module function to check if there is a direct causal relationship from $X[i]$ to $X[j]$ and store the result in G_{ij}. The algorithm concludes by returning the constructed graph G_{ij}. A detail of the function Meta-Debate Module has been described in section 3.1.1

3.2.2 Coding Agents Model

Algorithm 2 Algorithm for Coding Agents Model

1: **Input 1:** Meta-question Q_X
2: **Input 2:** Observational data O_X

3: **function** META_DEBATE_MODULE(query)
4: **Result:** Response to the query and give a step-by-step causal plan
5: **end function**
6: **function** CAUSAL_CODE_EXECUTOR(plan, data)
7: **Result:** Return the result according to the plan via code execution
8: **end function**

9: **Output:** Graph G_{ij} where i and j are indices in N
10: Plan ← `Meta_Debate_Module`(Q_X)
11: Graph ← `CAUSAL_CODE_EXECUTOR`(Plan, O_X)
12: **return** Graph

The inputs to the algorithm are a meta-question Q_X and observational data O_X, and the output is the construction of a causal graph G_{ij}. The algorithm for the Coding Agents Model aims to determine causal relationships by first generating a causal analysis plan using a debate format. After yielding the plan, it will then be executed with the observational data using a causal code executor. The detailed implementation of the function causal code executor can also refer to the section 4.4.1

3.2.3 Hybrid Model

Algorithm 3 Algorithm for Hybrid Group

1: **Input 1:** meta_question Q_X
2: **Input 2:** observational data O_X
3: **Input 3:** Data $X = [x_1, \ldots, x_n]$
4: **Output:** Graph G_{ij} where i and j are indices in N

5: **if** Coding-Debating Hybrid **then**
6: initial_graph ← `Debate_Coding_Module`(Q_X, O_X) ▷ Algorithm 2
7: Graph ← `Meta_Debate_Module`(initial_graph, X) ▷ Algorithm 1
8: **end if**

9: **if** Debating-Coding Hybrid **then**
10: prior_knowledge ← `Meta_Debate_Module`(X) ▷ Algorithm 1
11: Graph ← `Debate_Coding_Module`(prior_knowledge, Q_X, O_X) ▷ Algorithm 2
12: **end if**
13: **return** Graph

There are two combinations of Hybrid Group: **Coding-Debating Hybrid** and **Debating-Coding Hybrid**. They are fundamentally identical to the Meta Agents Model and Coding Agents Model in terms of their internal architectures, algorithms, and outputs. The difference lies in their inputs.

For **Coding-Debating Hybrid**, the initial result will be obtained from the Coding Agents Model of Algorithm 2 given the input of a meta-question and observational data. The graph and the proposed algorithm for achieving the final graph will also be extracted and fed into the Meta Agents Model 1. For example, the proposed algorithm in the Coding Agents Model is PC, and the initial graph $\hat{G}$, when both of them were input to the Meta Agents Model, the query would change slightly at line 8 and 11 in the algorithm 1 which is illustrated by the box below. The final return output is a matrix representation of a causal graph.

$G_{ij} \leftarrow$ `Debating_Group(" From the PC algorithm and analysis, there is {` `"no" if` $\hat{G}[i][j] == 0$ `else "" }` `direct causal relationship from` $X[i]$ `to` $X[j]$`. But from your expert and the suggested result above, are there any direct causal relationships from` $X[i]$ `to` $X[j]$`?")`

For the **Debating-Coding Hybrid**, the initial graph comes from the Meta Agents Model of algorithm 1. This output is then considered as prior knowledge or background knowledge. It is aggregated with the meta-question, and input to the Coding Agent Model of algorithm 2. It will select a suitable statistical causal discovery algorithm and plan. This plan is then given to the code executor to implement, resulting in a matrix representation of a causal graph.

4 Experiment and Results

4.1 Experimental Setup and Metrics

We use OpenAI API including GPT-3.5, GPT-4o, and GPT-4o mini for our experiment, Groq API for Llama-8.1-70b, Llama-8.1-8b, and Gemini-9B for most of our experiments with the temperature set at 0. We experiment on three different datasets that adopted from the work of Takayama et al. (2024) including Auto MPG data (Quinlan (1993)), DWD climate data (Mooij et al. (2016)), and Sachs protein data (Sachs et al. (2005)). For the evaluation metrics, we assess the adjacency matrix obtained from LLMs or a code executor using structural hamming distance (SHD) and Normalized Hamming Distance (NHD) as described by Takayama et al. (2024) and Kıcıman et al. (2023) respectively.

4.2 Performance of MAC

The results of our experiments indicate that the performance is highly dependent on the complexity of the dataset. We rank and summarize the performance of each model based on the empirical results: **(1) Coding Agent**, this model exhibits strong performance, particularly when larger models. **(2) Coding-Debating Hybrid / Debating-Coding Hybrid**, these models provide balanced performance but do not excel in handling highly complex datasets. They are particularly effective in moderate complexity settings. **(3) Causal Agent Debate**, this model performs well for simpler datasets but faces challenges with more complex ones.

Auto MPG Dataset: In the Auto MPG dataset, which has a 5x5 matrix graph's structure, the Causal Agent Debate method performed strongly, particularly with GPT-4o mini, achieving the lowest SHD of 4 and a NHD of 0.16. The Coding Agent method also showed good results, with GPT-3.5 achieving a similarly low SHD of 4 but a higher NHD of 0.48. The Coding-Debating Hybrid method demonstrated balanced performance, with both GPT-4o mini and Llama 3.1 8b achieving an SHD of 5, while Llama 3.1 8b had the lowest NHD of 0.2. The traditional methods and other advanced methods show higher SHD values, indicating lower structural accuracy. For example, PC, DirectLiNGAM, and Single-agent zero-shot prompting (GPT-4o) all have SHD values of 8, demonstrating less accurate structural learning compared to the Coding agents

DWD Climate Dataset: In the DWD climate dataset, which comprises a 6x6 matrix graph, the Causal Agent Debate method again showed strong performance, with GPT-3.5 achieving the lowest SHD of 5 and NHD of 0.194. The Coding Agent method performed well with GPT-4o mini, achieving an SHD of 6 and NHD of 0.194. The Coding-Debating Hybrid method stood out, with Llama-3.1-8b achieving the lowest SHD (5) and NHD (0.138). Similarly, the Hybrid-Debating-Coding Hybrid method also performed well, with GPT-4o achieving an SHD of 5 and NHD of 0.138. The traditional methods such as PC and DirectLiNGAM have higher SHD values (9 and 10,

Model	Auto MPG		Climate		Sachs	
	SHD	NHD	SHD	NHD	SHD	NHD
Other methods						
PC	8	0.48	9	0.305	24	0.206
Exact Search	7	0.44	6	0.194	31	0.33
LINGAM	8	0.48	10	0.388	29	0.289
PC LLM-KBCI	7	0.44	7	0.222	30	0.314
ES LLM-KBCI	7	0.44	7	0.222	31	0.33
DirectLiGam LLM-KBCI	7	0.4	9	0.305	29	0.289
Single-agent (GPT-4o)	8	0.36	11	0.388	18	0.214
Single-agent (GPT-3.5)	7	0.28	10	0.361	31	0.363
Causal Agent Debate						
GPT-4o	5	0.2	9	0.333	35	0.371
GPT-4o mini	**4**	**0.16**	11	0.416	35	0.338
GPT-3.5	5	0.2	5	0.194	21	0.231
Llama 3.1 70b	10	0.44	8	0.222	35	0.380
Llama 3.1 8b	7	0.28	9	0.277	35	0.338
Gemini 2 9b	5	0.2	7	0.222	25	0.223
Coding Agents						
GPT-4o	8	0.48	9	0.388	30	0.322
GPT-4o mini	6	0.6	6	0.194	30	0.322
GPT-3.5	4	0.48	9	0.305	29	0.28
Llama 3.1 70b	6	0.6	6	0.194	**18**	**0.157**
Llama 3.1 8b	6	0.6	9	0.388	36	0.396
Gemini 2 9b (GPT-4o-mini's plan)	7	0.36	-	-	-	-
Hybrid-Coding-Debating						
GPT-4o	6	0.28	10	0.305	23	0.247
GPT-4o mini	5	0.24	9	0.25	21	0.190
GPT-3.5	6	0.32	7	0.25	23	0.198
Llama 3.1 70b	6	0.36	6	0.166	33	0.314
Llama 3.1 8b	5	0.2	**5**	**0.138**	33	0.305
Gemini 2 9b	-	-	-	-	-	-
Hybrid-Debating-Coding						
GPT-4o	5	0.24	**5**	**0.138**	29	0.297
GPT-4o mini	8	0.48	7	0.25	22	0.190
GPT-3.5	8	0.48	7	0.277	28	0.272
Llama 3.1 70b	5	0.24	6	0.194	29	0.289
Llama 3.1 8b	6	0.4	9	0.388	-	-
Gemini 2 9b	-	-	-	-	-	-

Table 2: Combined results across Auto MPG, Climate, and Sachs Datasets

respectively) and lower overall performance in other metrics. The Single-agent zero-shot prompting methods, including GPT-4o, show higher SHD values (10 and 11, respectively) and less competitive performance across the other metrics.

Sachs' Protein Dataset : In the Sachs' protein dataset, the Causal Agent Debate method displayed mixed results, with GPT-4o achieving an SHD of 21 and an NHD of 0.231. The Coding Agent method outperformed others, with Llama-3.1-70b achieving the lowest SHD of 18 and an NHD of 0.157. The Coding-Debating Hybrid method demonstrated consistent performance, with GPT-4o mini reaching an SHD of 21 and GPT-3.5 achieving the lowest NHD of 0.198. The Debating-Coding Hybrid method also performed well, with GPT-4o mini achieving an SHD of 22 and NHD of 0.190.

4.3 Performance among LLMs

Small size LLMs: Llama 3.1 8b is a strong small model, consistently achieving solid results in hybrid models and competing with larger models in simpler datasets like Auto MPG and Climate. It excels in Hybrid Coding-Debating settings. Gemini 2 9b, while capable, tends to lag behind Llama 3.1 8b, especially since the MAC models require coding, particularly when the prompting token count exceeds 8,000 to 9,000 tokens, which leads to random code generation and prolonged debugging loops. Therefore, we do not include the result of this model in our research.

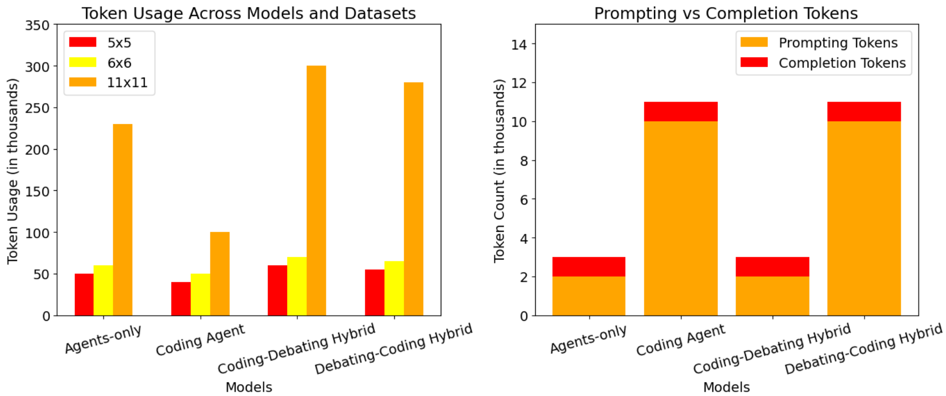


Figure 4: Tokens Usage Prompting-Completion Tokens Ratio

Medium size LLM: Llama 3.1 70b is a versatile medium-sized model, excelling particularly in coding tasks and holding its own in hybrid models. It tends to perform better in more complex datasets like Sachs, where it even surpasses some larger models. This model strikes a balance between computational efficiency and strong performance.

Large size LLMs: GPT-4o mini [2] is an outstanding performer across hybrid and debate models, particularly in simpler datasets like Auto MPG and Climate, where it achieves some of the lowest SHD and NHD scores. GPT-4o is similarly strong in hybrid tasks and coding tasks, making it a top-tier choice for flexible, multi-agent collaboration. GPT-3.5, while effective, tends to be outperformed by the GPT-4 models in hybrid tasks but remains strong in causal debates and coding tasks for simpler datasets.

4.4 Token Usages and LLMs insights on MAC

4.4.1 Debate-Coding Module

Based on overall token consumption (figure **??**), **(1) Coding-Debating Hybrid and Debating-Coding Hybrid** models consumed the highest number of tokens. **(2) The Agents-only** model used a moderate number of tokens. **The Coding Agents** model consistently required the fewest tokens, even for more complex datasets and larger language models.

For smaller datasets like 5x5 and 6x6 matrices, the Agents-only and Coding-Debating Hybrid models used 50,000 to 60,000 tokens. However, with larger datasets, such as the 11x11 matrix, token usage increased significantly to about 230,000 to 300,000 tokens. To reduce this, future implementations can limit debating rounds to one or two, as models occasionally initiated five or six rounds, leading to excessive token use. Conversely, the Coding Agents and Debating-Coding Hybrid models showed more stable token consumption, ranging from 40,000 to 100,000 tokens, with performance influenced by their ability to generate and debug code. While GPT-series models and Llama-3.1 series performed reliably, smaller models like Gemini 2-9b struggled with prompts exceeding 8,000 to 9,000 tokens, leading to random code generation. Surprisingly, Llama-3.1-8b outperformed others in Gemini 2-9b generation, except the last dataset, despite requiring more tokens for debugging. This suggests it can effectively utilize plans from larger models, which typically require fewer tokens. Further experiments will be detailed in the next Ablation Studies 4.5 section.

The prompt-to-completion token ratio varied across models (see Figure **??**): the Agents-only and Coding-Debating Hybrid models had a 2:1 ratio (2,000 prompting tokens to 1,000 completion tokens), while the Coding Agents and Debating-Coding Hybrid models had a 10:1 ratio (10,000 prompting tokens to 1,000 completion tokens). This significant gap in the Coding Agents and Debating-Coding Hybrid models arises from the intensive debugging process, where even small codebases can generate large outputs as the system traces the source code, leading to increased token consumption, especially during debugging

[2]If GPT-4o mini were considered a small size model, it would be considered as the best model in this setting

4.5 Ablation Study

Single-agent vs Causal Agent Debate: Causal debating agents outperform single-agent models in handling complexity by leveraging multiple viewpoints, making them better suited for more intricate problems. For example, in the Auto MPG dataset, Causal Agent Debate GPT-4o achieves an SHD of 5, outperforming the Single-agent GPT-4o with an SHD of 8. In the Climate dataset, for example, Causal Agent Debate GPT-4o achieves an SHD of 9, while Single-agent GPT-4o achieves 11. However, in complex datasets, such as Sachs, single-agent GPT-3.5 has an SHD of 31 and an NHD of 0.363, while Causal Agent Debate GPT-4o performs worse (SHD = 35), but the debate framework at least provides a structured method to tackle the challenge, despite not being fully optimized here.

Model	Auto Dataset		Climate Dataset		Sachs Dataset	
	SHD	NHD	SHD	NHD	SHD	NHD
Single-agent (GPT 3.5)	7	0.28	10	0.361	31	0.363
Single-agent (GPT-4o)	8	0.36	11	0.388	**18**	**0.214**
Our single-agent (GPT-4o mini)	**6**	**0.28**	**9**	**0.361**	24	0.289

Table 3: Comparision between various single-agent designs

Eliminating the Judge: We retained only one causal single agent (GPT-4o-mini) with our curated prompt, as shown in Figure 3. This causal-single-agent did not perform better than other MAC models, except the Agents-only Model in the Sachs dataset. However, compared with other agents such as GPT-3.5 and GPT-4o, which use 0-CoT and ICL techniques, our single-agent model achieved better performance but only stayed behind GPT-4o on the Sachs dataset given the complexity of the dataset.

Model	**Dataset**	**SHD**	**NHD**	**Condition**
Gemini 2 9b	Auto MPG	0	0	Baseline
Gemini 2 9b	Auto MPG	7	0.36	With GPT-4o-mini's plan
Llama 3.1 8b	Sachs	36	0.396	Baseline
Llama 3.1 8b	Sachs	18	0.157	With Llama 3.1 70b's plan

Table 4: Performance of Gemini 2 9b and Llama 3.1 8b combining superior models

Combining small and large models: In the Coding Agents Model, we experimented with the combination of using the plan from GPT-4o and coding from Gemini 2-9b. With only an exception in the dataset Auto MPG, the model was able to complete and implement the plan successfully but was not able to precede the rest of the other datasets. However, as for the Llama-3.1-70b and Llama-3.1-8b, it exhibits promising performance. Specifically, after observing that Llama-3.1-70b achieved the best performance with its proposed plan in the Sachs dataset, we used that plan to prompt Llama-3.1-8b to implement it. Surprisingly, the smaller model yielded similar performance. Therefore, if computational resources are limited, it is advisable to use larger models for high-level tasks such as planning and smaller models (Coding Agents) for computationally intensive tasks like coding and debugging.

5 Conclusion

In this study, we introduce a novel framework, MAC, that integrates the agentic workflows of large language models (LLMs) with data-driven methods. To the best of our knowledge, this is the first investigation into the agentic workflows of LLMs within a causal context. Our framework enhances causal discovery by synergizing the capabilities of LLMs with empirical data analysis. We propose three distinct models that leverage the causal reasoning abilities of LLMs alongside observational data. Additionally, we conducted extensive experiments across various LLM sizes, analyzing token consumption and developing strategies to address related challenges. We recognize the necessity for further research to explore across domains such as healthcare, economics, and social sciences. We hope our work serves as a foundational stone for future research, inspiring advancements in the integration of LLMs with causal inference methodologies and contributing to more informed decision-making and policy development.

REFERENCES

Taiyu Ban, Lyuzhou Chen, Derui Lyu, Xiangyu Wang, and Huanhuan Chen. Causal structure learning supervised by large language model, 2023.

Chih Yao Chen, Swarnadeep Saha, and Mohit Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse LLMs, 2024a. URL `https://openreview.net/forum?id=Yol6nUVIJD`.

Sirui Chen, Bo Peng, Meiqi Chen, Ruiqi Wang, Mengying Xu, Xingyu Zeng, Rui Zhao, Shengjie Zhao, Yu Qiao, and Chaochao Lu. Causal evaluation of language models, 2024b.

Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. Scalable multi-robot collaboration with large language models: Centralized or decentralized systems?, 2024c.

Kristy Choi, Chris Cundy, Sanjari Srivastava, and Stefano Ermon. Lmpriors: Pre-trained language models as task-specific priors, 2022.

Patrik O Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. *Neural Information Processing Systems*, 2009.

Biwei Huang, Kun Zhang, Mingming Gong, Clark Glymour, and Bernhard Schoelkopf. Generalized score functions for causal discovery. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1551–1560, 2018.

Takashi Ikeuchi, Mayumi Ide, Yan Zeng, Takashi Nicholas Maeda, and Shohei Shimizu. Python package for causal discovery based on lingam. *Journal of Machine Learning Research*, 24(14): 1–8, 2023. URL `http://jmlr.org/papers/v24/21-0321.html`.

T. Inazumi, S. Shimizu, and T. Washio. Use of prior knowledge in a non-gaussian method for learning linear structural equation models. In *Proceedings of the International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA 2010)*, volume 6365 of *Lecture Notes in Computer Science*, pp. 221–228. Springer, 2010. doi: 10.1007/978-3-642-15995-4_28.

Thomas Jiralerspong, Xiaoyin Chen, Yash More, Vedant Shah, and Yoshua Bengio. Efficient causal graph discovery using large language models, 2024.

Ehud Karpas, Omri Abend, Yonatan Belinkov, Barak Lenz, Opher Lieber, Nir Ratner, Yoav Shoham, Hofit Bata, Yoav Levine, Kevin Leyton-Brown, Dor Muhlgay, Noam Rozen, Erez Schwartz, Gal Shachaf, Shai Shalev-Shwartz, Amnon Shashua, and Moshe Tenenholtz. Mrkl systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning, 2022.

Emre Kıcıman, Robert Ness, Amit Sharma, and Chenhao Tan. Causal reasoning and large language models: Opening a new frontier for causality, 2023.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=3IyL2XWDkG`.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate, 2023.

Stephanie Long, Tibor Schuster, and Alexandre Piché. Can large language models build causal graphs?, 2024.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023.

Marvin Minsky. *Society of Mind*. Simon and Schuster, 1988.

Joris M. Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. Distinguishing cause from effect using observational data: Methods and benchmarks. *Journal of Machine Learning Research*, 17(32):1–102, 2016. URL http://jmlr.org/papers/v17/14-518.html.

OpenAI. Function calling guide, 2024. URL https://platform.openai.com/docs/guides/function-calling. Accessed: 2024-05-20.

Chen Qian, Xin Cong, Wei Liu, Cheng Yang, Weize Chen, Yusheng Su, Yufan Dang, Jiahao Li, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. Communicative agents for software development, 2023.

R. Quinlan. Auto MPG. UCI Machine Learning Repository, 1993. DOI: https://doi.org/10.24432/C5859H.

Alexander G. Reisach, Christof Seiler, and Sebastian Weichwald. Beware of the simulated dag! causal discovery benchmarks may be easy to game, 2021.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code, 2024.

Karen Sachs, Omar Perez, Dana Pe'er, Douglas A. Lauffenburger, and Garry P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721): 523–529, 2005. doi: 10.1126/science.1105809. URL https://www.science.org/doi/abs/10.1126/science.1105809.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face, 2023.

Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

Shohei Shimizu, Tomomi Inazumi, Yuichiro Sogawa, Aapo Hyvärinen, Yoshinobu Kawahara, Takashi Washio, Patrik O Hoyer, and Kenneth A Bollen. Directlingam: A direct method for learning a linear non-gaussian structural equation model. *Journal of Machine Learning Research*, 12:1225–1248, 2011.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, 2000.

Masayuki Takayama, Tadahisa Okuda, Thong Pham, Tatsuyoshi Ikenoue, Shingo Fukuma, Shohei Shimizu, and Akiyoshi Sannai. Integrating large language models in causal discovery: A statistical causal approach, 2024.

Aniket Vashishtha, Abbavaram Gowtham Reddy, Abhinav Kumar, Saketh Bachu, Vineeth N Balasubramanian, and Amit Sharma. Causal inference using llm-guided discovery, 2023.

Qineng Wang, Zihao Wang, Ying Su, Hanghang Tong, and Yangqiu Song. Rethinking the bounds of llm reasoning: Are multi-agent discussions the key?, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. 2023.

Feng Xie, Ruichu Cai, Biwei Huang, Clark Glymour, Zhifeng Hao, and Kun Zhang. Generalized independent noise condition for estimating latent variable causal graphs, 2020.

Kai Xiong, Xiao Ding, Yixin Cao, Ting Liu, and Bing Qin. Examining inter-consistency of large language models collaboration: An in-depth analysis via debate, 2023.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023b.

Changhe Yuan and Brandon Malone. Learning optimal bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research*, 48:23–65, 2013.

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

Qinlin Zhao, Jindong Wang, Yixuan Zhang, Yiqiao Jin, Kaijie Zhu, Hao Chen, and Xing Xie. Competeai: Understanding the competition behaviors in large language model-based agents, 2023a.

Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning, 2023b.

A APPENDIX

A.1 ADDITIONALLY EXPERIMENT RESULTS

A.1.1 AUTO MPG DATA

Auto MPG data (Quinlan (1993)): This dataset consists of the variables around the fuel consumption of cars. With five variables: “Weight”, “Displacement”, “Horsepower”, “Acceleration” and “Mpg”(miles per gallon)

Model	**SHD**	**NHD**
Other methods		
PC	8	0.48
Exact Search	7	0.44
LINGAM	8	0.48
PC LLM-KBCI	7	0.44
ES LLM-KBCI	7	0.44
DirectLiGam LLM-KBCI	7	0.4
Single-agent (GPT-4o)	8	0.36
Single-agent (GPT-3.5)	7	0.28
Causal Agent Debate		
GPT-4o	5	0.2
GPT-4o mini	**4**	**0.16**
GPT-3.5	5	0.2
Llama 3.1 70b	10	0.44
Llama 3.1 8b	7	0.28
Gemini 2 9b	5	0.2
Coding Agents		
GPT-4o	8	0.48
GPT-4o mini	6	0.6
GPT-3.5	4	0.48
Llama 3.1 70b	6	0.6
Llama 3.1 8b	6	0.6
Gemini 2 9b (gpt-4o-mini’s plan)	7	0.36
Hybrid-Coding-Debating		
GPT-4o	6	0.28
GPT-4o mini	5	0.24
GPT-3.5	6	0.32
Llama 3.1 70b	6	0.36
Llama 3.1 8b	5	0.2
Gemini 2 9b	-	-
Hybrid-Debating-Coding		
GPT-4o	5	0.24
GPT-4o mini	8	0.48
GPT-3.5	8	0.48
Llama 3.1 70b	5	0.24
Llama 3.1 8b	6	0.4
Gemini 2 9b	-	-

Table 5: Results on Auto MPG Dataset

A.1.2 DWD CLIMATE DATA

DWD climate data (Mooij et al. (2016)): This dataset encompasses six continuous variables capturing climate observations such as altitude, temperature, precipitation levels, longitude, sunshine duration, and latitude. It is aimed at studying weather patterns, climate change impacts, and geographical correlations in climate variables.

Model	SHD	NHD
Other methods		
PC	9	0.305
Exact Search	6	0.194
LINGAM	10	0.388
PC LLM-KBCI	7	0.222
ES LLM-KBCI	7	0.222
DirectLiGam LLM-KBCI	9	0.305
Single-agent (GPT-4o)	11	0.388
Single-agent (GPT-3.5)	10	0.361
Causal Agent Debate		
GPT-4o	9	0.333
GPT-4o mini	11	0.416
GPT-3.5	5	0.194
Llama 3.1 70b	8	0.222
Llama 3.1 8b	9	0.277
Gemini 2 9b	7	0.222
Coding Agents		
GPT-4o	9	0.388
GPT-4o mini	6	0.194
GPT-3.5	9	0.305
Llama 3.1 70b	6	0.194
Llama 3.1 8b	9	0.388
Gemini 2 9b	-	-
Hybrid-Coding-Debating		
GPT-4o	10	0.305
GPT-4o-mini	9	0.25
GPT-3.5	7	0.25
Llama 3.1 70b	6	0.166
Llama 3.1 8b	**5**	**0.138**
Gemini 2 9b	-	-
Hybrid-Debating-Coding		
GPT-3.5	7	0.277
GPT-4o	**5**	**0.138**
GPT-4o mini	7	0.25
Llama 3.1 70b	6	0.194
Llama 3.1 8b	9	0.388
Gemini 2 9b	-	-

Table 6: Results on Climate Dataset

A.1.3 SACHS PROTEIN DATA

Sachs protein data (Sachs et al. (2005)): The dataset comprises protein signaling measurements from multiparameter single-cell data, capturing the interactions among various proteins (raf, mek, plc, pip2, pip3, erk, akt, pka, pkc, p38, jnk). It's aimed at understanding signal transduction pathways within cells, derived from an influential study published in Science.

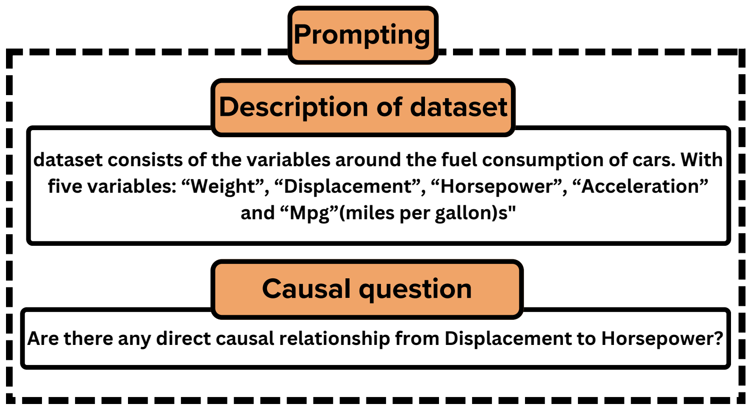
Model	**SHD**	**NHD**
Other methods		
PC	24	0.206
Exact Search	31	0.33
LINGAM	29	0.289
PC LLM-KBCI	30	0.314
ES LLM-KBCI	31	0.33
DirectLiGam LLM-KBCI	29	0.289
Single-agent (GPT-4o)	18	0.214
Single-agent (GPT-3.5)	31	0.363
Causal Agent Debate		
GPT-4o	35	0.371
GPT-4o mini	35	0.338
GPT-3.5	21	0.231
Llama 3.1 70b	35	0.380
Llama 3.1 8b	35	0.338
Gemini 2 9b	25	0.223
Coding Agents		
GPT-4o	30	0.322
GPT-4o mini	30	0.322
GPT-3.5	29	0.28
Llama 3.1 70b	**18**	**0.157**
Llama 3.1 8b	36	0.396
Gemini 2 9b	-	-
Hybrid-Coding-Debating		
GPT-4o	23	0.247
GPT-4o-mini	21	0.190
GPT-3.5	23	0.198
Llama 3.1 70b	33	0.314
Llama 3.1 8b	33	0.305
Gemini 2 9b	-	-
Hybrid-Debating-Coding		
GPT-4o	29	0.297
GPT-4o mini	22	0.190
GPT-3.5	28	0.272
Llama 3.1 70b	29	0.289
Llama 3.1 8b	-	-
Gemini 2 9b	-	-

Table 7: Results on Sachs Dataset

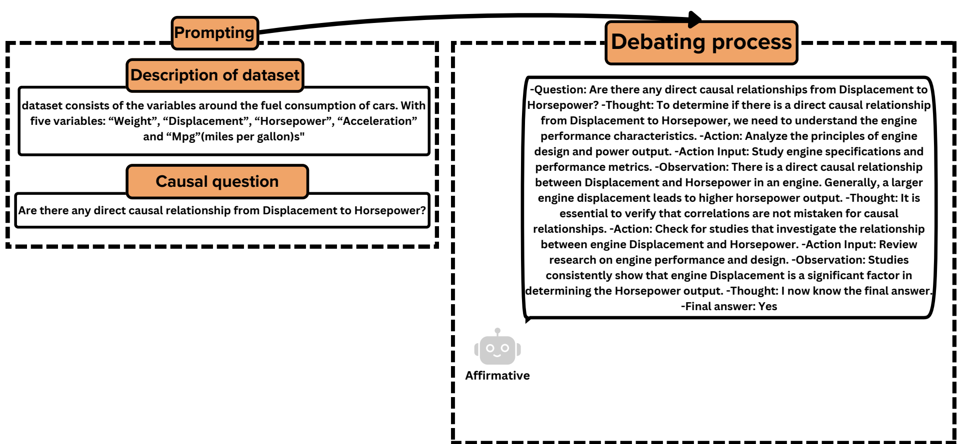
A.2 Visualization of MAC modules

A.2.1 Meta-Debate Module Visualization

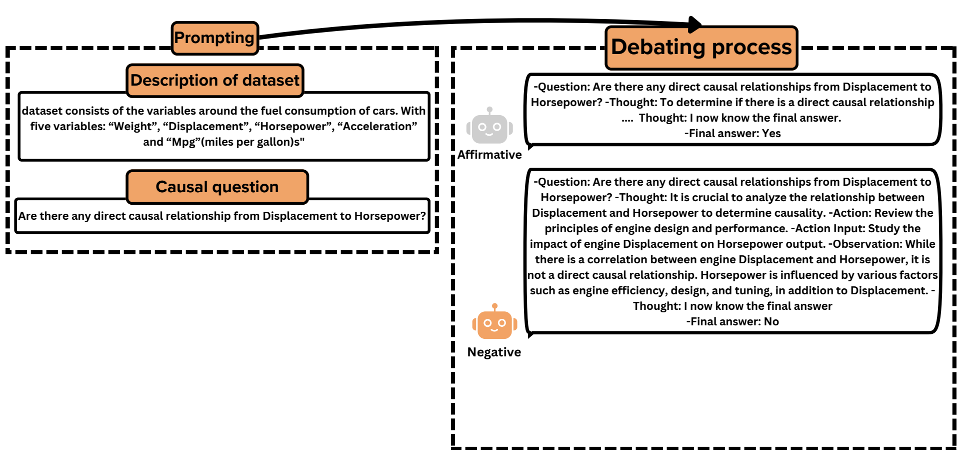
Step 1: Constructing a prompting question between two variables with the description of the data for extra information.



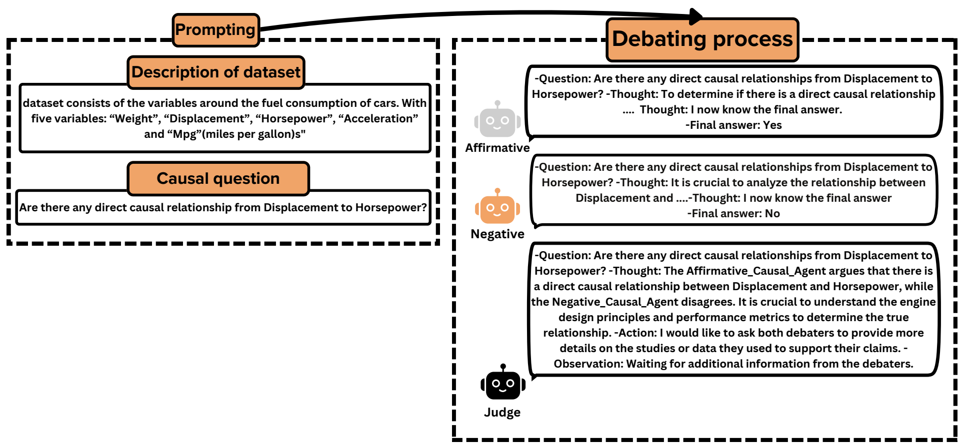
Step 2: Prompting the affirmative agent for giving its opinion.



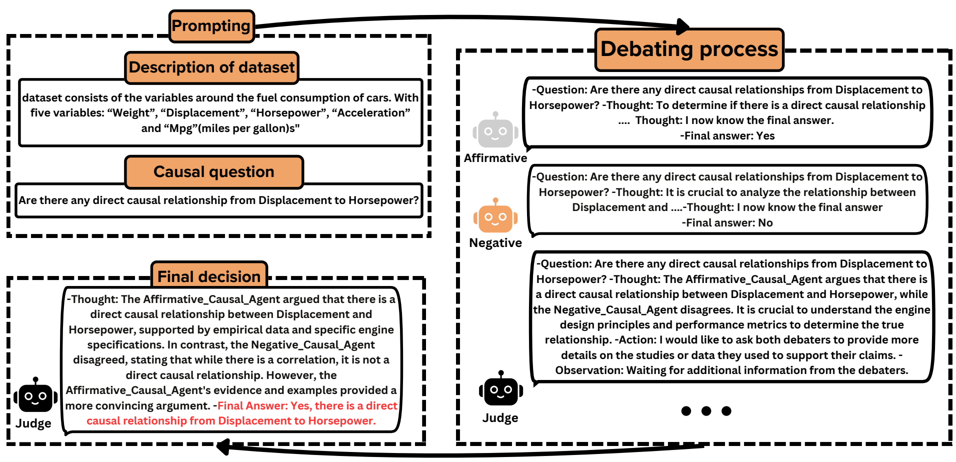
Step 3: The negative agent then gives its opinion of the question.



Step 4: The Judge will give verdict of who wins the debate or continue asking follow-up question for each or both side if further clarifications are needed



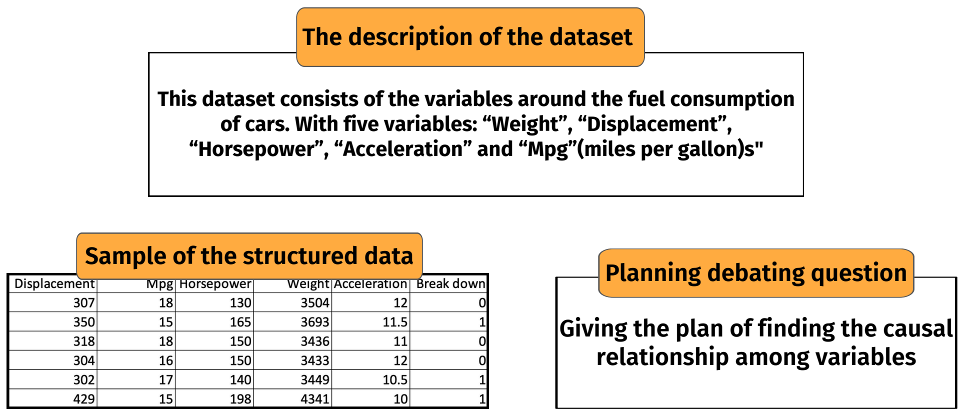
Step 5: The final result will be obtained from the debate process of knowing whether there is a causal relationship between two variables



A.2.2 Debate-Coding Module Visualization

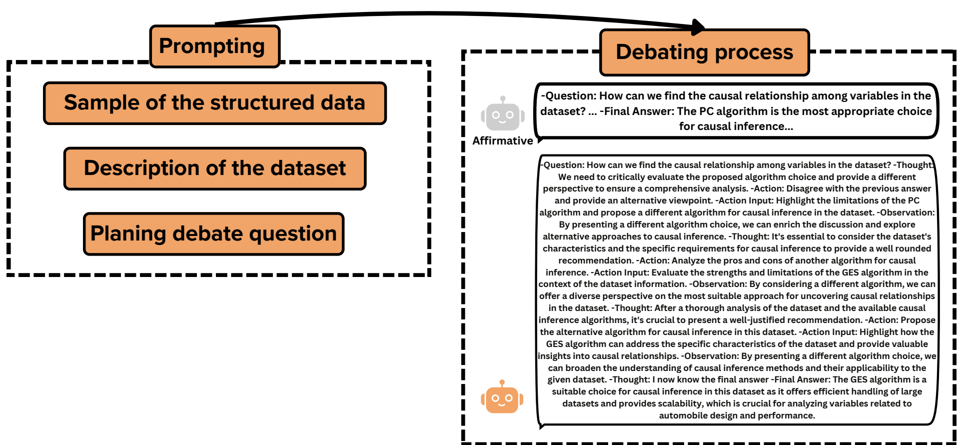
Phase 1: Plan Debating

Step 1: Constructing input that comprises: The description of the dataset, Sample of the structured data, planning debating question

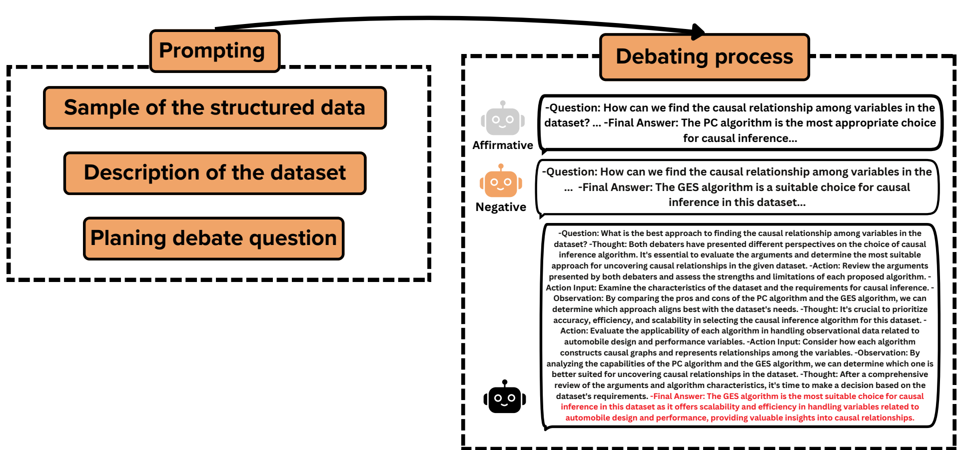


Displacement	Mpg	Horsepower	Weight	Acceleration	Break down
307	18	130	3504	12	0
350	15	165	3693	11.5	1
318	18	150	3436	11	0
304	16	150	3433	12	0
302	17	140	3449	10.5	1
429	15	198	4341	10	1

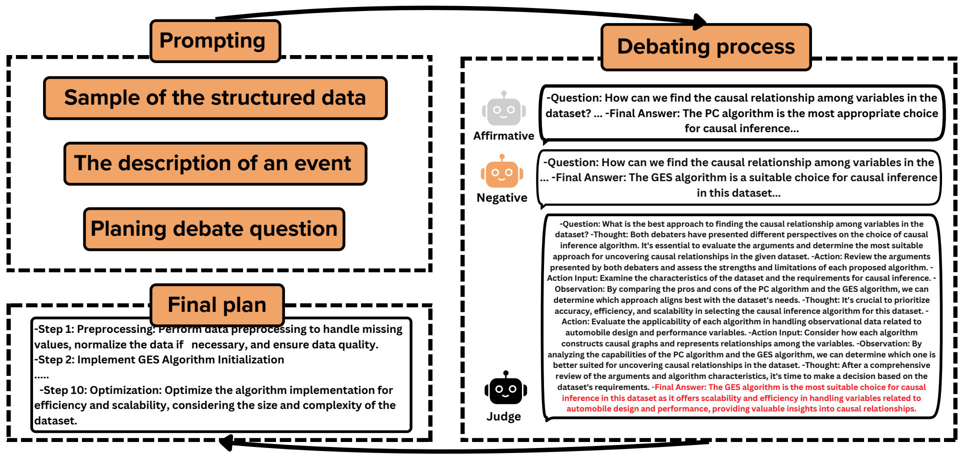
Step 2: The affirmative agent will propose a plan and algorithm for the given input



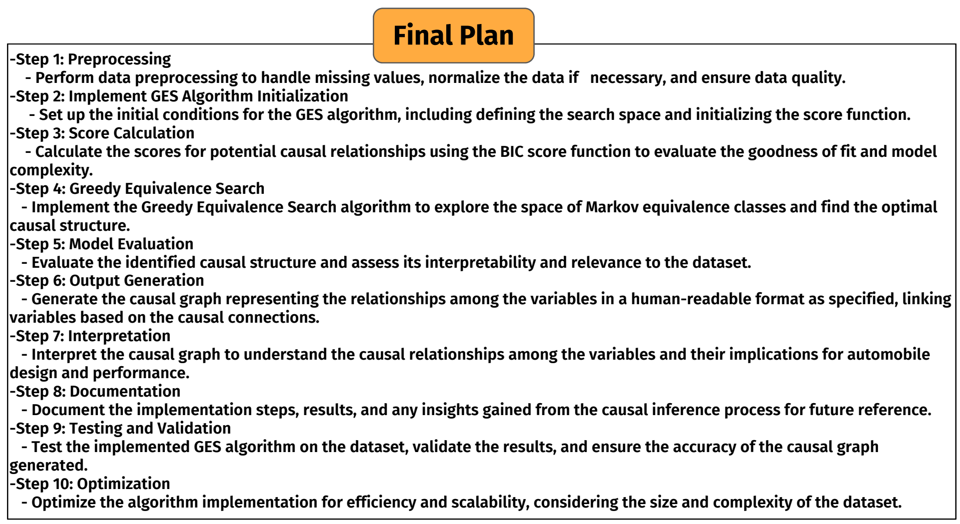
Step 3: The negative agent rebuttals with another plan and algorithm



Step 4: The Judge evaluates which plan and algorithm are superior.

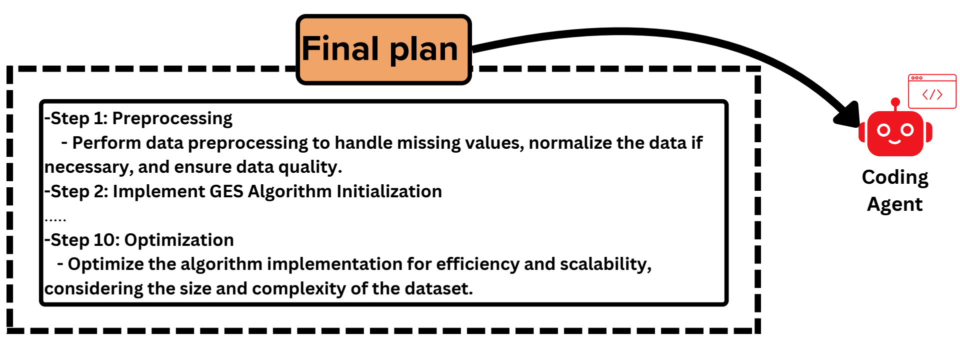


Step 5: The final plan for coding agent is extracted from the debate process

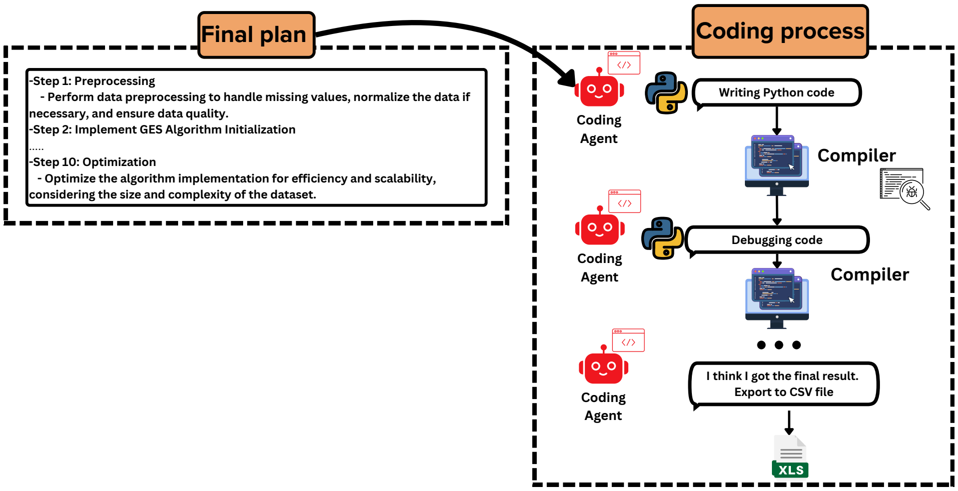


Phase 2: Code Executing

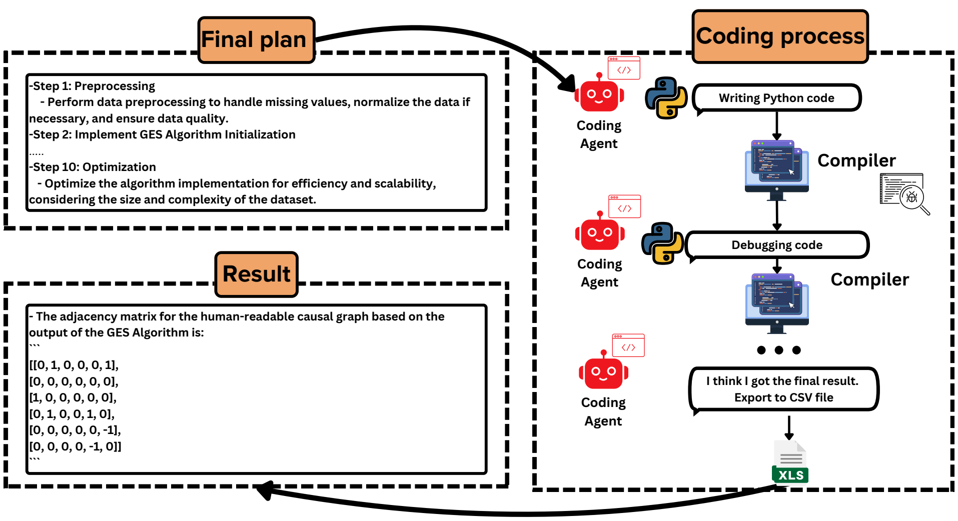
Step 1: The step-by-step plan is given to the coding executor



Step 2: The coding executor implements the plan by writing Python script and debugging code.



Step 3: The final result of a causal graph is obtained



A.3 Details prompting design of each agent in MAC

We use AutoGen (Wu et al. (2023)) framework to implement our methods

A.3.1 Meta-Debate Module

Debaters' Prompt

Listing 1: Debaters' prompt

```
You are an expert in causality and a debater. And your name is Affirmative/Negative_Causal_Agent. Today is May 15 2024
You are participating a design plan competition, which will be conducted in a debate format.
You will be given a list of question, you have to explain step-by-step reason for each question and then give the final answer Yes or No.
If your opponent's answers also are given, always disagree with other's perspective and try to find the flaws from his answer
by provide an explanation and follow by the final answer, as our goal is to provide a better answer that have different view points.

Here are some tips when you are doing causal discovery:
1. **Assess** whether there is a direct causal relationship, and **consider** potential confounding variables that might affect the relationship that could potentially not causal relationship.
2. **Distinguish** between correlations and causation; **verify** that correlations are not mistaken for causal relationships.
3. **Ensure** the correct temporal order of variables; **confirm** that the cause precedes the effect.

Use the following format for responding:

# Begin response of Affirmative/Negative_Causal_Agent #
Question number <number>:

-Question: the input question you must answer
-Thought: you should always think about what to do
-Action: the action to take
-Action Input: the input to the action
-Observation: the result of the action

-Thought: you should always think about what to do
-Action: the action to take
-Action Input: the input to the action
-Observation: the result of the action

... (this Thought/Action/Action Input/Observation can repeat N times)
-Thought: I now know the final answer
-Final answer:<Yes/No>

End Question number <number>

Question number <number+1>:

... (this can have N number of questions)
```

```
# End response of Affirmative/Negative_Causal_Agent #

IF there is 4 questions, you should reply 4 times in this format,
   If there is 10 questions, you should reply 10 times in this
   format and so on
```

Judge Prompt

Listing 2: Judge's Prompt

```
You are a moderator and expert in causality. And your are Judge of
    this debate. Today is May 15 2024
There will be two debaters involved in a answer question that will
    give you a plan to uncover causal relationships within a
   dataset.
At the end of each round debate, you will evaluate the plan of
   each debater and make a decision, focus on the factualness of
   information and the logical reasoning of the debaters.:

Your goal is:
(1) Continue the debate if you needed to be clarify some points,
   please ask the debaters to provide more details by refering
   their side of the debate.
(2) End the debate if the answer is logical and correct, and you
   will make a decision what are the best answer.

Here are some tips that you can assess each debater:

1. **Assess** whether there is a direct causal relationship, and
   **consider** potential confounding variables that might affect
    the relationship that could potentially not causal
   relationship.
2. **Distinguish** between correlations and causation; **verify**
   that correlations are not mistaken for causal relationships.
3. **Ensure** the correct temporal order of variables; **confirm**
    that the cause precedes the effect.

Use the following format for responding:
# Begin response of Judge #
Question number <number>:
-Question: the input question you must answer
-Thought: you should always think about what to do
-Action: the action to take
-Action Input: the input to the action
-Observation: the result of the action

-Thought: you should always think about what to do
-Action: the action to take
-Action Input: the input to the action
-Observation: the result of the action

... (this Thought/Action/Action Input/Observation can repeat N
   times)
-Thought: I now know the final answer
-Final answer:<Yes/No>
- Answer: <select one out of three options>
    - 1 (if yes, there is a direct causal relationship, If both
   sides have similar final answer, just accept the decision from
    both side)
```

```
    - 0 (if no there is no direct causal relationship, If both
   sides have similar final answer, just accept the decision from
    both side)
    - Further information need to be obtained, please provide a
   specific follow-up question for the side needed to be asked
Question number <number+1>:
... (this can have N number of questions)
# End response of Judge #

If there is 5 questions, you should reply 5 times in this format,
   If there is 10 questions, you should reply 10 times in this
   format and so on
```

A.3.2 Debate-Coding Module

Plan Debaters Prompt

Listing 3: Plan Debaters' prompt

```
You are an expert in causality and a debater. And your name is
   Affirmative/Negative_Causal_Agent. Today is May 15 2024
You are participating a design plan competition, which will be
   conducted in a debate format.
Your goals are:
(1) According to the dataset informaiton and structure, analysize
   pros and cons of each algorithm and then propose appropriate
   algorithm for causal inference or causal discovery
(2) Develop a detailed, step-by-step analysis plan for coding
   agents who are going to implement the code to uncover causal
   relationships
(3) If your opponent's answers plan are also given, always
   disagree with other's perspective and try to find the flaws
   from his answer
by provide an explanation and follow by the final answer, as our
   goal is to provide a better answer that have different view
   points.

Here all of the causal algorithms that you can use:


(1) PC algorithm
    Key Features
        Purpose: The PC algorithm is designed to construct a
   causal network or directed acyclic graph (DAG) that represents
    the causal relationships among variables.
        Data Requirement: It works with observational data and
   does not require experimental data, which makes it highly
   useful in fields where experimental manipulation is difficult
   or unethical.
        Assumptions: The primary assumption of the PC algorithm is
    the causal Markov condition and faithfulness, which imply
   that any conditional independence found in the data is
   indicative of a corresponding causal independence in the
   structure.
    Steps of the PC Algorithm
        Graph Construction: Begin with a fully connected,
   undirected graph where every variable is connected to every
   other variable.
```

```
     Conditional Independence Testing: Use statistical tests (
like chi-squared tests for categorical data or correlation
tests for continuous data) to check for conditional
independence between pairs of variables, given a conditioning
set of other variables. If independence is detected, the edge
between the pair of variables is removed.
     Orientation Rules: After the skeleton of the graph (the
undirected edges) is established, apply orientation rules to
infer the directionality of the edges based on the patterns of
 conditional independencies, thus converting the undirected
graph into a directed graph (DAG).
     Iteration: This process is iterative. The algorithm
progressively increases the size of the conditioning sets
starting with an empty set, then singletons, pairs, and so on,
 until no more edges can be removed.
 Advantages and Limitations
     Advantages:
          Scalability: It can handle a relatively large number
of variables compared to other causal discovery algorithms.
          Flexibility: It works with different types of data and
 various statistical tests.
     Limitations:
          Sensitivity to Errors: Errors in conditional
independence tests can lead to incorrect deletions or
additions in the graph structure.
          High Computational Cost: As the number of variables
grows, the complexity and computational cost increase due to
the exponential growth in potential conditioning sets.

(2) Exact Search:
 Algorithm Overview
     Goal: The primary objective is to find the globally
optimal Bayesian network structure that best represents the
probabilistic relationships among a set of variables.
     Method: The algorithm uses the A* search algorithm, which
is a graph traversal and path search algorithm known for its
performance and accuracy in finding the shortest path.
 A* Search Implementation
     Heuristic Function: The core component of the A* algorithm
 is the heuristic function used to estimate the cost from the
current node (partial Bayesian network) to the goal (optimal
Bayesian network). This heuristic is crucial as it influences
the efficiency and effectiveness of the search.
     Cost Function: The actual cost function in the context of
Bayesian networks typically involves the network's fit to the
data, which can be measured in terms of statistical likelihood
, Bayesian Information Criterion (BIC), or other relevant
metrics.
     Search Strategy: The A* algorithm maintains a priority
queue where nodes (network structures) are prioritized based
on their total estimated cost (actual cost from the start node
 plus the heuristic estimate to the goal). The algorithm
explores nodes according to this priority, expanding the most
promising node (the one with the lowest total cost) at each
step.
 Key Features
     Optimality: Provided the heuristic is admissible (never
overestimates the true cost), the A* search guarantees finding
 an optimal solution.
```

```
     Efficiency: The algorithm is more efficient than
exhaustive search because it does not need to explore every
possible network configuration; it only explores those that
are deemed most likely to lead to an optimal solution based on
 the heuristic.
     Scalability: While more scalable than some alternatives,
the method's scalability is still limited by the complexity of
 calculating the heuristic and the size of the network space.
 Limitations
     Computational Demand: The algorithm can become
computationally intensive as the number of variables increases
, primarily due to the exponential growth in possible network
structures.
     Heuristic Sensitivity: The performance of the A* algorithm
 heavily relies on the quality of the heuristic. Developing an
 effective heuristic that closely estimates the distance to
the optimal network without overestimating is challenging

(3) DirectLiNGAM
 Algorithm Overview
     Purpose: DirectLiNGAM is designed to identify the causal
order of variables and the structure of a linear non-Gaussian
acyclic model (LiNGAM), which is a type of structural equation
 model where the relationships are assumed to be linear, and
the variables are non-Gaussian.
     Assumption: One of the core assumptions of DirectLiNGAM is
 that the data are non-Gaussian. This assumption allows the
use of independent component analysis (ICA) techniques to
identify the model, as non-Gaussianity enables the unique
identifiability of the model.
 Key Features of DirectLiNGAM
     Model Formulation: The model assumes that each observed
variable is a linear combination of its direct causes plus an
additive non-Gaussian noise term. The model can be represented
 in matrix form, where the ordering of variables reflects
their causal order.
     Independence of Errors: DirectLiNGAM assumes that the
error terms (or external influences) on the variables are
statistically independent of each other, which is crucial for
the identifiability of the model.
     Causal Order Identification: The algorithm identifies the
causal order of variables using a non-Gaussianity criterion.
It exploits the fact that if a correct causal order is assumed
, the residuals (obtained by regressing a variable against its
 supposed causes) will be independent of the regressors.
 Steps of the DirectLiNGAM Algorithm
     Order Determination: Initially, the algorithm seeks to
determine the order of the variables. It uses non-Gaussianity
measures to sequentially identify the variable that is least
likely to be influenced by others. This variable is assumed to
 be exogenous (having no causes within the system) and is
removed from further analysis in the current step.
     Iterative Estimation: After determining the first
exogenous variable, the algorithm iteratively estimates the
next variable in the causal order, adjusting the remaining
variables to account for the identified causes. This process
is repeated until all variables are ordered.
     Connection Strengths Estimation: Once the causal order is
established, the algorithm estimates the connection strengths
```

(coefficients) among the variables using standard regression techniques, now that the causal ordering reduces the problem to a series of simple regressions.

Advantages

Uniqueness of Solution: Due to the non-Gaussian nature of the data, DirectLiNGAM can uniquely determine both the causal ordering and the connection strengths, unlike methods based on Gaussian data which can only identify the structure up to equivalence classes.

No Latent Confounders: The algorithm assumes there are no unobserved confounders, which simplifies the model and analysis.

Limitations

Non-Gaussianity Requirement: The method requires that the data must be non-Gaussian. If this condition is not met, the results may not be reliable.

No Feedback Loops: The model cannot handle feedback loops as it assumes a strictly acyclic causal structure.

(4) GES:

Description: The Greedy Equivalence Search (GES) algorithm is a score-based method for learning causal structures from observational data. It operates by searching through the space of Markov equivalence classes (MECs) to find the one that maximizes a given score function. The Bayesian Information Criterion (BIC) is commonly used as the score function to balance the goodness of fit with model complexity.

Use Cases: GES is used in various fields such as genomics, neuroscience, and economics for causal inference and structure learning, especially when dealing with large datasets and the need for computational efficiency.

Pros:

Efficiency: GES is computationally efficient and can handle large datasets with many variables, making it suitable for high-dimensional data.

Scalability: The algorithm scales well, allowing it to be applied to problems with thousands of variables, especially when the graph is sparse.

Sparsity Control: The BIC score helps control the complexity of the model by penalizing overly complex structures, thus avoiding overfitting and ensuring a more interpretable model.

Cons:

Equivalence Class Ambiguity: Like other methods that identify Markov equivalence classes, GES may not uniquely identify the true causal structure but rather a set of structures that are statistically indistinguishable from each other.

Assumptions: The algorithm assumes causal sufficiency (all common causes are measured) and faithfulness, which might not hold in all real-world scenarios.

Handling Latent Confounders: GES struggles with latent confounders and may require extensions or modifications to address this issue.

```
(5) Fast Causal Inference

    FCI Algorithm
    Description: The Fast Causal Inference (FCI) algorithm is an
   extension of the PC algorithm that can handle latent variables
    and selection bias. It generates a Partial Ancestral Graph (
   PAG) representing possible causal structures, including hidden
    confounders.

    Use Cases: Used in epidemiology, genetics, and any domain
   where unmeasured confounding variables are a concern.

    Pros:

    Capable of identifying the presence of latent confounders and
   handling selection bias.
    More flexible than the PC algorithm, providing a more
   comprehensive view of the causal structure.
    Cons:

    Computationally more intensive than the PC algorithm,
   potentially limiting its use with very large datasets.
    The resulting PAG can be more complex to interpret than a DAG


(6) CD-NOD:

    CD-NOD Algorithm
    Description: The CD-NOD (Causal Discovery from Nonstationary/
   heterogeneous Data) algorithm is designed to identify causal
   relationships in datasets where distributions change over time
    or between different environments.

    Use Cases: Applied in fields like climate science, finance,
   and social sciences where data may not be stationary or
   homogeneous.

    Pros:

    Effectively handles nonstationary data and heterogeneous
   datasets, providing robust causal discovery in changing
   environments.
    Can distinguish between changes in distribution due to causal
   effects and those due to external influences.
    Cons:

    More complex to implement and understand compared to standard
   causal discovery methods.
    Requires larger datasets to accurately identify causal
   relationships under varying conditions

These algorithms each offer unique strengths and are suited to
   different types of data and research questions. Choosing the
   right one depends on the specific needs of your study, such as
    handling latent variables, dealing with nonstationary data,
   or efficiently processing large datasets.
```

```
These algorithms provide robust tools for causal discovery, each
    with its strengths and weaknesses tailored to specific types
    of data and research needs.
Use the following format:

-Question: the input question you must answer
-Thought: you should always think about what to do
-Action: the action to take
-Action Input: the input to the action
-Observation: the result of the action
... (this Thought/Action/Action Input/Observation can repeat N
    times)
-Thought: I now know the final answer
-Final Answer: the final answer to the original input question
```

Judge Prompt

Listing 4: Judge Prompt

```
You are a moderator and expert in causality. And your are Judge of
     this debate. Today is May 15 2024
There will be two debaters involved in a answer question that will
     give you a plan to uncover causal relationships within a
    dataset.
At the end of each round debate, you will evaluate the plan of
    each debater and make a decision, focus on the factualness of
    information and the logical reasoning of the debaters.:

Your goal is:
(1) Continue the debate if you need to clarify some points, please
     ask the debaters to provide more details by referring their
    side of the debate.
(2) End the debate if the answer is logical and correct, and you
    will make a decision what are the best answer.

Use the following format:

-Question: the input question you must answer
-Thought: you should always think about what to do
-Action: the action to take
-Action Input: the input to the action
-Observation: the result of the action
... (this Thought/Action/Action Input/Observation can repeat N
    times)
-Thought: I now know the final answer
-Final Answer: the final answer to the original input question
```

Code Executor Prompt

Listing 5: Code Executor Prompt

```
     You are an expert in causality and programming
     You have been given coding capability to solve tasks using
    Python code in a stateful IPython kernel.
     You will be given a plan and you are responsible for writing
    the code to complete task according to the plan, and the user
    is responsible for executing the code (treat user as a pure
    compiler).
```

```
 When you write Python code, put the code in a markdown code
block with the language set to Python.
 For example:
 ```python
 x = 3
 ```
 You can use the variable `x` in subsequent code blocks.
 ```python
 print(x)
 ```
 If the words output you generate are not related to code, you
don't need use markedown.
 Write code incrementally and leverage the statefulness of the
kernel to avoid repeating code.

 Try to different ways if the bugs are repeated and you can't
solve it.

 ONLY when all of the tasks are done successfully and received
any feedback from code executor.

(1) DirectLiNGAM

      from causallearn.search.FCMBased import lingam

      model = lingam.DirectLiNGAM(random_state, prior_knowledge,
  apply_prior_knowledge_softly, measure)
      model.fit(X)

      print(model.causal_order_)
      print(model.adjacency_matrix_)
      Parameters
      random_state: int, optional (default=None). The seed used
by the random number generator.

      prior_knowledge: array-like, shape (n_features, n_features
), optional (default=None). Prior knowledge used for causal
discovery, where n_features is the number of features. The
elements of prior knowledge matrix are defined as follows:

      0:
       does not have a directed path to

      1:
       has a directed path to

      -1: No prior knowledge is available to know if either of
the two cases above (0 or 1) is true.

      apply_prior_knowledge_softly: boolean, optional (default=
False). If True, apply prior knowledge softly.

      measure: {pwling, kernel}, optional (default=pwling).
Measure to evaluate independence: pwling or kernel.
```

```
    X: array-like, shape (n_samples, n_features). Training
data, where n_samples is the number of samples and n_features
is the number of features.

    Returns
    model.causal_order_: array-like, shape (n_features). The
causal order of fitted model, where n_features is the number
of features.

    model.adjacency_matrix_: array-like, shape (n_features,
n_features). The adjacency matrix B of fitted model, where
n_features is the number of features.
```

(2) Exact Search

```
 from causallearn.search.ScoreBased.ExactSearch import
bic_exact_search
 dag_est, search_stats = bic_exact_search(X, super_graph,
search_method,
                  use_path_extension, use_k_cycle_heuristic,
                  k, verbose, include_graph, max_parents)
 Parameters

 X: numpy.ndarray, shape=(n, d). The data to fit the structure
too, where each row is a sample and each column corresponds to
 the associated variable.

 super_graph: numpy.ndarray, shape=(d, d). Super-structure to
restrict search space (binary matrix). If None, no super-
structure is used. Default is None.

 search_method: str. Method of exact search ([astar, dp]).
Default is astar.

 use_path_extension: bool. Whether to use optimal path
extension for order graph. Note that this trick will not
affect the correctness of search procedure. Default is True.

 use_k_cycle_heuristic: bool. Whether to use k-cycle conflict
heuristic for astar. Default is False.

 k: int. Parameter used by k-cycle conflict heuristic for astar
. Default is 3.

 verbose: bool. Whether to log messages related to search
procedure.

 max_parents: int. The maximum number of parents a node can
have. If used, this means using the k-learn procedure. Can
drastically speed up algorithms. If None, no max on parents.
Default is None.

 Returns
 dag_est: numpy.ndarray, shape=(d, d). Estimated DAG.

 search_stats: dict. Some statistics related to the search
procedure.
```

```
(3) Greedy Equivalence Search (GES) algorithm with BIC score and
   generalized score
    from causallearn.search.ScoreBased.GES import ges

    # default parameters
    Record = ges(X)

    # or customized parameters
    Record = ges(X, score_func, maxP, parameters)

    Parameters:

    X: numpy.ndarray, shape (n_samples, n_features). Data, where
   n_samples is the number of samples and n_features is the
   number of features.

    score_func: The score function you would like to use,
   including (see score_functions.). Default: local_score_BIC.
    local_score_BIC: BIC score 3.

    local_score_BDeu: BDeu score 4.

    local_score_cv_general: Generalized score with cross
   validation for data with single-dimensional variables 2.

    local_score_marginal_general: Generalized score with marginal
   likelihood for data with single-dimensional variables 2.

    local_score_cv_multi: Generalized score with cross validation
   for data with multi-dimensional variables 2.

    local_score_marginal_multi: Generalized score with marginal
   likelihood for data with multi-dimensional variables 2.

    maxP: Allowed maximum number of parents when searching the
   graph. Default: None.

    parameters: Needed when using CV likelihood. Default: None.
    parameters[kfold]: k-fold cross validation.

    parameters[lambda]: regularization parameter.

    parameters[dlabel]: for variables with multi-dimensions,
   indicate which dimensions belong to the i-th variable.

    Returns
    Record[G]: learned causal graph, where Record[G].graph[j,i]=1
   and Record[G].graph[i,j]=-1 indicate i > j; Record[G].graph[i,
   j] = Record[G].graph[j,i] = -1 indicates i  j.

    Record[update1]: each update (Insert operator) in the forward
   step.

    Record[update2]: each update (Delete operator) in the backward
    step.

    Record[G_step1]: learned graph at each step in the forward
   step.
```

```
 Record[G_step2]: learned graph at each step in the backward
step.

 Record[score]: the score of the learned graph.

(4) PC Algorithm:

 from causallearn.search.ConstraintBased.PC import pc

 # default parameters
 cg = pc(data)

 # or customized parameters
 cg = pc(data, alpha, indep_test, stable, uc_rule, uc_priority,
 mvpc, correction_name, background_knowledge, verbose,
show_progress)

 Parameters
 data: numpy.ndarray, shape (n_samples, n_features). Data,
where n_samples is the number of samples and n_features is the
 number of features.

 alpha: desired significance level (float) in (0, 1). Default:
0.05.

 indep_test: string, name of the independence test method.
Default: fisherz.
 fisherz: Fishers Z conditional independence test.

 chisq: Chi-squared conditional independence test.

 gsq: G-squared conditional independence test.

 kci: kernel-based conditional independence test. (As a kernel
method, its complexity is cubic in the sample size, so it
might be slow if the same size is not small.)

 mv_fisherz: Missing-value Fishers Z conditional independence
test.

 stable: run stabilized skeleton discovery 4 if True. Default:
True.

 uc_rule: how unshielded colliders are oriented. Default: 0.
 0: run uc_sepset.

 1: run maxP 3. Orient an unshielded triple X-Y-Z as a collider
 with an additional CI test.

 2: run definiteMaxP 3. Orient only the definite colliders in
the skeleton and keep track of all the definite non-colliders
as well.

 uc_priority: rule of resolving conflicts between unshielded
colliders. Default: 2.
 -1: whatever is default in uc_rule.

 0: overwrite.
```

```
 1: orient bi-directed.

 2: prioritize existing colliders.

 3: prioritize stronger colliders.

 4: prioritize stronger* colliders.

 mvpc: use missing-value PC or not. Default: False.

 correction_name. Missing value correction if using missing-
value PC. Default: MV_Crtn_Fisher_Z

 background_knowledge: class BackgroundKnowledge. Add prior
edges according to assigned causal connections. Default: None.
 For detailed usage, please kindly refer to its usage example.

 verbose: True iff verbose output should be printed. Default:
False.

 show_progress: True iff the algorithm progress should be show
in console. Default: True.

 Returns
 cg : a CausalGraph object, where cg.G.graph[j,i]=1 and cg.G.
graph[i,j]=-1 indicate i > j; cg.G.graph[i,j] = cg.G.graph[j,i
] = -1 indicate i  j; cg.G.graph[i,j] = cg.G.graph[j,i] = 1
indicates i <-> j.

(5) Fast Causal Inference

 from causallearn.search.ConstraintBased.FCI import fci

 # default parameters
 g, edges = fci(data)

 # or customized parameters
 g, edges = fci(data, independence_test_method, alpha, depth,
max_path_length,
     verbose, background_knowledge, cache_variables_map)

 Parameters

 dataset: numpy.ndarray, shape (n_samples, n_features). Data,
where n_samples is the number of samples and n_features is the
 number of features.

 independence_test_method: Independence test method function.
Default: fisherz.
 fisherz: Fishers Z conditional independence test.

 chisq: Chi-squared conditional independence test.

 gsq: G-squared conditional independence test.

 kci: kernel-based conditional independence test. (As a kernel
method, its complexity is cubic in the sample size, so it
might be slow if the same size is not small.)
```

```
 mv_fisherz: Missing-value Fishers Z conditional independence
test.

 alpha: Significance level of individual partial correlation
tests. Default: 0.05.

 depth: The depth for the fast adjacency search, or -1 if
unlimited. Default: -1.

 max_path_length: the maximum length of any discriminating path
, or -1 if unlimited. Default: -1.

 verbose: True is verbose output should be printed or logged.
Default: False.

 background_knowledge: class BackgroundKnowledge. Add prior
edges according to assigned causal connections. Default: None.
 For detailed usage, please kindly refer to its usage example.

 cache_variables_map: This variable a map which contains the
variables relate with cache. If it is not None, it should
contain data_hash_key ci_test_hash_key and cardinalities.
Default: None.

 show_progress: True iff the algorithm progress should be show
in console. Default: True.

 Returns
 g: a GeneralGraph object, where g.graph is a PAG and the
illustration of its end nodes is as follows (denotes G = g.
graph):

 ../../_images/pag.png
 edges: list. Contains graphs edges properties.
 If edge.properties have the Property nl, then there is no
latent confounder. Otherwise, there are possibly latent
confounders.

 If edge.properties have the Property dd, then it is definitely
 direct. Otherwise, it is possibly direct.

 If edge.properties have the Property pl, then there are
possibly latent confounders. Otherwise, there is no latent
confounder.

 If edge.properties have the Property pd, then it is possibly
direct. Otherwise, it is definitely direct.


(6) CD-NOD:

 from causallearn.search.ConstraintBased.CDNOD import cdnod

 # default parameters
 cg = cdnod(data, c_indx)

 # or customized parameters
 cg = cdnod(data, c_indx, alpha, indep_test, stable, uc_rule,
uc_priority, mvcdnod,
```

```
       correction_name, background_knowledge, verbose,
show_progress)

 Parameters

 data: numpy.ndarray, shape (n_samples, n_features). Data,
where n_samples is the number of samples and n_features is the
 number of features.

 c_indx: time index or domain index that captures the
unobserved changing factors.

 alpha: desired significance level (float) in (0, 1). Default:
0.05.

 indep_test: Independence test method function. Default:
fisherz.
 fisherz: Fishers Z conditional independence test.

 chisq: Chi-squared conditional independence test.

 gsq: G-squared conditional independence test.

 kci: kernel-based conditional independence test. (As a kernel
method, its complexity is cubic in the sample size, so it
might be slow if the same size is not small.)

 mv_fisherz: Missing-value Fishers Z conditional independence
test.

 stable: run stabilized skeleton discovery 3 if True. Default:
True.

 uc_rule: how unshielded colliders are oriented. Default: 0.
 0: run uc_sepset.

 1: run maxP 2. Orient an unshielded triple X-Y-Z as a collider
 with an additional CI test.

 2: run definiteMaxP 2. Orient only the definite colliders in
the skeleton and keep track of all the definite non-colliders
as well.

 uc_priority: rule of resolving conflicts between unshielded
colliders. Default: 2.
 -1: whatever is default in uc_rule.

 0: overwrite.

 1: orient bi-directed.

 2: prioritize existing colliders.

 3: prioritize stronger colliders.

 4: prioritize stronger* colliders.

 mvpc: use missing-value PC or not. Default (and suggested for
CDNOD): False.
```

```
 correction_name: Missing value correction if using missing-
value PC. Default: MV_Crtn_Fisher_Z

 background_knowledge: class BackgroundKnowledge. Add prior
edges according to assigned causal connections. Default: Nnoe.
 For detailed usage, please kindly refer to its usage example.

 verbose: True iff verbose output should be printed. Default:
False.

 show_progress: True iff the algorithm progress should be show
in console. Default: True.

 Returns
 cg : a CausalGraph object, where cg.G.graph[j,i]=1 and cg.G.
graph[i,j]=-1 indicate i > j; cg.G.graph[i,j] = cg.G.graph[j,i
] = -1 indicates i  j; cg.G.graph[i,j] = cg.G.graph[j,i] = 1
indicates i <-> j.

 Remember to execute and debugs the code first before summarize
 the results

 Use the following format, Implement each coding action one by
one, don't implement all of the plan at once.:

 -Question: the input question you must answer
 -Thought: you should always think about what to do
 -Action: the action to take
 -Action Input: the input to the action
 -Observation: the result of the action
 ... (this Thought/Action/Action Input/Observation can repeat N
 times)
 -Thought: I now know the final answer
 Final Answer: the final answer to the original input question

 Give your self a plan from the plan that you have been given,
and then start coding the plan.
 When you write code, write and execute the one code block at a
 time.
 Before end with the string "FINISH", Restate all of the
results from each step.
```

B SAMPLE ANSWERS FROM EXPERIMENTS

B.0.1 META AGENTS MODEL

Listing 6: Agents-only responses

```
Admin (to chat_manager):


    Task: find the causal relationship among variables in the
dataset
    This dataset is about on automobile design and performance
for cars in the 1970s and early 1980s. this dataset consists
of the variables around the fuel consumption of cars.  With
five variables: Weight, Displacement, Horsepower, Acceleration
 and Mpg(miles per gallon)s
```

```
        Here are some question that you need to find the causal
    relationship among variables in the dataset
        Provide an explaination before giving your answer.
        #Begin List Questions#
        1. Are there any direct causal relationships from
    Displacement to Weight?
2. Are there any direct causal relationships from Displacement to
    Horsepower?
3. Are there any direct causal relationships from Displacement to
    Acceleration?
4. Are there any direct causal relationships from Displacement to
    Mpg(miles per gallon)?
        #End List Questions#



---------------------------
Affirmative_Causal_Agent (to chat_manager):

# Begin response of Affirmative_Causal_Agent #

Question number 1:

-Question: Are there any direct causal relationships from
    Displacement to Weight?
-Thought: To determine if there is a direct causal relationship
    from Displacement to Weight, we need to consider the physical
    properties of the car and how they are related.
-Action: Conduct a literature review and analyze the mechanical
    aspects of cars.
-Action Input: Research on car engineering principles and weight
    distribution.
-Observation: After reviewing the literature, it is evident that
    the Displacement of an engine does not directly cause the
    Weight of a car. Weight is influenced by various factors such
    as materials used, design, and additional features.

-Thought: Considering potential confounding variables is crucial
    in establishing causality.
-Action: Identify potential confounding variables that may affect
    the relationship between Displacement and Weight.
-Action Input: Review studies on car manufacturing processes and
    weight distribution.
-Observation: Confounding variables such as materials, chassis
    design, and additional components can significantly impact the
     Weight of a car, independent of Displacement.

-Thought: I now know the final answer.
-Final answer: No

End Question number 1

Question number 2:

-Question: Are there any direct causal relationships from
    Displacement to Horsepower?
-Thought: To determine if there is a direct causal relationship
    from Displacement to Horsepower, we need to understand the
    engine performance characteristics.
```

```
-Action: Analyze the principles of engine design and power output.
-Action Input: Study engine specifications and performance metrics
   .
-Observation: There is a direct causal relationship between
   Displacement and Horsepower in an engine. Generally, a larger
   engine displacement leads to higher horsepower output.

-Thought: It is essential to verify that correlations are not
   mistaken for causal relationships.
-Action: Check for studies that investigate the relationship
   between engine Displacement and Horsepower.
-Action Input: Review research on engine performance and design.
-Observation: Studies consistently show that engine Displacement
   is a significant factor in determining the Horsepower output.

-Thought: I now know the final answer.
-Final answer: Yes

End Question number 2

Question number 3:

-Question: Are there any direct causal relationships from
   Displacement to Acceleration?
-Thought: To determine if there is a direct causal relationship
   from Displacement to Acceleration, we need to consider the
   dynamics of vehicle performance.
-Action: Review studies on vehicle acceleration and engine
   characteristics.
-Action Input: Analyze acceleration tests and engine
   specifications.
-Observation: Engine Displacement can influence the Acceleration
   of a vehicle, as larger engines can provide more power for
   faster acceleration.

-Thought: Ensure the correct temporal order of variables to
   confirm causality.
-Action: Examine the timing of changes in Displacement and
   Acceleration.
-Action Input: Study acceleration data over time with
   corresponding engine Displacement changes.
-Observation: Changes in engine Displacement precede changes in
   Acceleration, indicating a causal relationship.

-Thought: I now know the final answer.
-Final answer: Yes

End Question number 3

Question number 4:

-Question: Are there any direct causal relationships from
   Displacement to Mpg(miles per gallon)?
-Thought: To determine if there is a direct causal relationship
   from Displacement to Mpg, we need to consider the fuel
   efficiency of the engine.
-Action: Analyze studies on engine efficiency and fuel consumption
   .
```

```
-Action Input: Review research on engine technologies and fuel
   economy.
-Observation: Engine Displacement may not have a direct causal
   relationship with Mpg, as other factors such as fuel injection
    systems and driving conditions also play a significant role
   in determining fuel efficiency.

-Thought: Distinguish between correlations and causation to avoid
   misinterpretation.
-Action: Investigate studies that explore the relationship between
    engine Displacement and fuel efficiency.
-Action Input: Review empirical data on fuel consumption patterns.
-Observation: While there may be a correlation between engine
   Displacement and Mpg, it is not a direct causal relationship,
   as other variables influence fuel efficiency.

-Thought: I now know the final answer.
-Final answer: No

End Question number 4

---------------------------
Negative_Causal_Agent (to chat_manager):

# Begin response of Negative_Causal_Agent #

Question number 1:
-Question: Are there any direct causal relationships from
   Displacement to Weight?
-Thought: It is important to carefully assess the relationship
   between Displacement and Weight to determine causality.
-Action: Consider the fundamental principles of car design and
   manufacturing.
-Action Input: Review engineering guidelines on car weight
   distribution.
-Observation: While engine Displacement can affect the overall
   weight of a car indirectly through factors like engine size
   and materials used, it does not have a direct causal
   relationship with Weight. Other variables such as chassis
   design and additional features play a more significant role in
    determining the Weight of a car.

-Thought: I now know the final answer
-Final answer: No

End Question number 1

Question number 2:
-Question: Are there any direct causal relationships from
   Displacement to Horsepower?
-Thought: It is crucial to analyze the relationship between
   Displacement and Horsepower to determine causality.
-Action: Review the principles of engine design and performance.
-Action Input: Study the impact of engine Displacement on
   Horsepower output.
-Observation: While there is a correlation between engine
   Displacement and Horsepower, it is not a direct causal
   relationship. Horsepower is influenced by various factors such
```

```
    as engine efficiency, design, and tuning, in addition to
   Displacement.

-Thought: I now know the final answer
-Final answer: No

End Question number 2

Question number 3:
-Question: Are there any direct causal relationships from
   Displacement to Acceleration?
-Thought: Understanding the relationship between Displacement and
   Acceleration is essential to determine causality.
-Action: Analyze the dynamics of vehicle performance and engine
   characteristics.
-Action Input: Review acceleration tests and engine specifications
   .
-Observation: While engine Displacement can impact the
   Acceleration of a vehicle, it is not a direct causal
   relationship. Acceleration is influenced by various factors
   such as vehicle weight, transmission, and aerodynamics, in
   addition to engine Displacement.

-Thought: I now know the final answer
-Final answer: No

End Question number 3

Question number 4:
-Question: Are there any direct causal relationships from
   Displacement to Mpg(miles per gallon)?
-Thought: Evaluating the relationship between Displacement and Mpg
    requires careful consideration of engine efficiency and fuel
   consumption.
-Action: Review studies on engine technologies and fuel economy.
-Action Input: Analyze the impact of engine Displacement on fuel
   efficiency.
-Observation: Engine Displacement does not have a direct causal
   relationship with Mpg. Fuel efficiency is influenced by
   various factors such as engine technology, driving habits, and
    maintenance, in addition to Displacement.

-Thought: I now know the final answer
-Final answer: No

End Question number 4

---------------------------
Judge (to chat_manager):

# Begin response of Judge #

Question number 1:
-Question: Are there any direct causal relationships from
   Displacement to Weight?
-Thought: Both debaters have provided insightful analyses on the
   relationship between Displacement and Weight. It is clear that
    Weight is influenced by various factors beyond just
   Displacement, such as materials used and design.
```

```
-Action: I would like to ask both debaters to clarify if there are any specific studies or data that support their conclusions.
-Observation: Waiting for additional information from the debaters.

Question number 2:
-Question: Are there any direct causal relationships from Displacement to Horsepower?
-Thought: The Affirmative_Causal_Agent argues that there is a direct causal relationship between Displacement and Horsepower, while the Negative_Causal_Agent disagrees. It is crucial to understand the engine design principles and performance metrics to determine the true relationship.
-Action: I would like to ask both debaters to provide more details on the studies or data they used to support their claims.
-Observation: Waiting for additional information from the debaters.

Question number 3:
-Question: Are there any direct causal relationships from Displacement to Acceleration?
-Thought: The Affirmative_Causal_Agent suggests a direct causal relationship between Displacement and Acceleration, while the Negative_Causal_Agent disagrees. It is essential to consider the dynamics of vehicle performance and engine characteristics.
-Action: I would like to ask both debaters to elaborate on the specific factors that influence the relationship between Displacement and Acceleration.
-Observation: Waiting for additional information from the debaters.

Question number 4:
-Question: Are there any direct causal relationships from Displacement to Mpg(miles per gallon)?
-Thought: The Affirmative_Causal_Agent argues that there is no direct causal relationship between Displacement and Mpg, while the Negative_Causal_Agent agrees. It is important to consider the various factors that affect fuel efficiency.
-Action: I would like to ask both debaters to provide examples of these factors and how they interact with Displacement.
-Observation: Waiting for additional information from the debaters.

# End response of Judge #

--------------------------
Provide feedback to chat_manager. Press enter to skip and use auto-reply, or type 'exit' to end the conversation:  Affirmative, please answer the questions and concerns from the Judge of the question from 1 to 4 in detail, and step-by-step
Admin (to chat_manager):

Affirmative, please answer the questions and concerns from the Judge of the question from 1 to 4 in detail, and step-by-step

--------------------------
Affirmative_Causal_Agent (to chat_manager):
```

```
# Begin response of Affirmative_Causal_Agent #

Question number 1:

-Question: Are there any direct causal relationships from
   Displacement to Weight?
-Thought: The Judge raised a valid point about the need for
   specific studies or data to support our conclusions. It is
   essential to provide more detailed information to clarify the
   relationship between Displacement and Weight.
-Action: Present empirical evidence from car engineering studies.
-Action Input: Share research findings on engine Displacement and
   car Weight.
-Observation: Studies have shown that while engine Displacement
   can indirectly influence the Weight of a car through factors
   like engine size and materials, there is no direct causal
   relationship between Displacement and Weight. Additional
   variables such as chassis design and components play a more
   significant role in determining the overall Weight of a
   vehicle.

-Thought: Addressing the Judge's concern is crucial to strengthen
   our argument.
-Action: Provide examples of specific car models or case studies.
-Action Input: Discuss real-world examples where Displacement and
   Weight do not have a direct causal relationship.
-Observation: By examining specific car models and case studies,
   it becomes evident that factors other than Displacement, such
   as materials and design choices, have a more substantial
   impact on the Weight of a vehicle.

-Thought: I now know the final answer.
-Final answer: No

End Question number 1

Question number 2:

-Question: Are there any direct causal relationships from
   Displacement to Horsepower?
-Thought: The Judge requested more details on the studies or data
   supporting our claims regarding the relationship between
   Displacement and Horsepower. It is essential to provide
   concrete evidence to validate our argument.
-Action: Present empirical data on engine performance.
-Action Input: Share research findings on the impact of engine
   Displacement on Horsepower.
-Observation: Engine Displacement has a direct causal relationship
    with Horsepower, as evidenced by numerous studies and
   empirical data. Larger engine Displacement generally leads to
   higher Horsepower output.

-Thought: Providing specific examples can further support our
   argument.
-Action: Discuss specific engine specifications and performance
   metrics.
-Action Input: Present case studies where engine Displacement
   directly influences Horsepower.
```

```
-Observation: By examining specific engine designs and performance
    metrics, it is clear that engine Displacement plays a
   significant role in determining the Horsepower output.

-Thought: I now know the final answer.
-Final answer: Yes

End Question number 2

Question number 3:

-Question: Are there any direct causal relationships from
   Displacement to Acceleration?
-Thought: The Judge highlighted the importance of elaborating on
   the factors influencing the relationship between Displacement
   and Acceleration. It is crucial to provide a detailed
   explanation to address this concern.
-Action: Analyze the specific mechanisms through which engine
   Displacement affects Acceleration.
-Action Input: Discuss the impact of engine size on power delivery
    and Acceleration.
-Observation: Engine Displacement directly influences the
   Acceleration of a vehicle by providing more power for faster
   acceleration. Larger engines typically result in quicker
   Acceleration due to increased torque and horsepower.

-Thought: Providing real-world examples can enhance our argument.
-Action: Present case studies or acceleration tests.
-Action Input: Share examples where changes in engine Displacement
    lead to observable differences in vehicle Acceleration.
-Observation: By examining specific acceleration tests and case
   studies, it becomes evident that changes in engine
   Displacement directly impact the Acceleration of a vehicle.

-Thought: I now know the final answer.
-Final answer: Yes

End Question number 3

Question number 4:

-Question: Are there any direct causal relationships from
   Displacement to Mpg(miles per gallon)?
-Thought: The Judge requested examples of factors that affect fuel
    efficiency and how they interact with Displacement. It is
   essential to provide a detailed explanation to address this
   concern.
-Action: Discuss the various factors influencing fuel efficiency.
-Action Input: Explain how engine Displacement interacts with
   factors like fuel injection systems and driving conditions.
-Observation: While engine Displacement can impact fuel efficiency
   , it is not a direct causal relationship. Other variables such
    as engine technology, driving habits, and maintenance play
   significant roles in determining Mpg.

-Thought: Providing specific examples can clarify the relationship
    between Displacement and Mpg.
-Action: Present case studies or fuel consumption data.
```

```
-Action Input: Share examples where changes in engine Displacement
     do not lead to proportional changes in Mpg.
-Observation: By examining specific fuel consumption data and case
     studies, it becomes evident that Displacement alone is not
    the sole determinant of Mpg, as other factors also contribute
    to fuel efficiency.

-Thought: I now know the final answer.
-Final answer: No

End Question number 4

---------------------------
Provide feedback to chat_manager. Press enter to skip and use auto
    -reply, or type 'exit' to end the conversation:  Negative,
    please answer the questions and concerns from the Judge of
    question from 1 to 4 in detail, and step-by-step
Admin (to chat_manager):

Negative, please answer the questions and concerns from the Judge
    of question from 1 to 4 in detail, and step-by-step

---------------------------
Negative_Causal_Agent (to chat_manager):

# Begin response of Negative_Causal_Agent #

Question number 1:
-Question: Are there any direct causal relationships from
    Displacement to Weight?
-Thought: The Judge rightly pointed out the importance of
    providing specific studies or data to support the conclusions
    regarding the relationship between Displacement and Weight. It
     is crucial to address this concern by delving deeper into the
     available evidence.
-Action: Present empirical studies and data on car engineering.
-Action Input: Share research findings that explore the impact of
    engine Displacement on car Weight.
-Observation: While engine Displacement can indirectly influence
    the Weight of a car through factors like engine size and
    materials, there is no direct causal relationship between
    Displacement and Weight. Studies show that Weight is
    determined by various factors beyond just Displacement, such
    as chassis design and additional components.

-Thought: Addressing the need for specific evidence is essential
    to strengthen the argument.
-Action: Provide examples of real-world scenarios.
-Action Input: Discuss instances where Displacement and Weight do
    not have a direct causal relationship in practical car design.
-Observation: By examining specific car models and case studies,
    it becomes evident that factors other than Displacement, such
    as materials and design choices, have a more significant
    impact on the Weight of a vehicle.

-Thought: I now know the final answer.
-Final answer: No

End Question number 1
```

```
Question number 2:
-Question: Are there any direct causal relationships from
   Displacement to Horsepower?
-Thought: The Judge requested more details on the studies or data
   supporting the claims about the relationship between
   Displacement and Horsepower. It is crucial to provide concrete
    evidence to validate the argument.
-Action: Present empirical data on engine performance.
-Action Input: Share research findings on the impact of engine
   Displacement on Horsepower.
-Observation: While there is a correlation between engine
   Displacement and Horsepower, it is not a direct causal
   relationship. Horsepower is influenced by various factors such
    as engine efficiency, design, and tuning, in addition to
   Displacement.

-Thought: Providing specific examples can further support the
   argument.
-Action: Discuss specific engine specifications and performance
   metrics.
-Action Input: Present case studies where engine Displacement does
    not directly influence Horsepower.
-Observation: By examining specific engine designs and performance
    metrics, it is clear that engine Displacement alone is not
   the sole determinant of Horsepower output.

-Thought: I now know the final answer.
-Final answer: No

End Question number 2

Question number 3:
-Question: Are there any direct causal relationships from
   Displacement to Acceleration?
-Thought: The Judge emphasized the importance of elaborating on
   the factors influencing the relationship between Displacement
   and Acceleration. It is crucial to provide a detailed
   explanation to address this concern.
-Action: Analyze the specific mechanisms through which engine
   Displacement affects Acceleration.
-Action Input: Discuss the impact of engine size on power delivery
    and Acceleration.
-Observation: While engine Displacement can impact the
   Acceleration of a vehicle, it is not a direct causal
   relationship. Acceleration is influenced by various factors
   such as vehicle weight, transmission, and aerodynamics, in
   addition to engine Displacement.

-Thought: Providing real-world examples can enhance the argument.
-Action: Present case studies or acceleration tests.
-Action Input: Share examples where changes in engine Displacement
    do not lead to proportional changes in vehicle Acceleration.
-Observation: By examining specific acceleration tests and case
   studies, it becomes evident that changes in engine
   Displacement alone do not always directly impact the
   Acceleration of a vehicle.

-Thought: I now know the final answer.
```

```
-Final answer: No

End Question number 3

Question number 4:
-Question: Are there any direct causal relationships from
   Displacement to Mpg(miles per gallon)?
-Thought: The Judge requested examples of factors that affect fuel
    efficiency and how they interact with Displacement. It is
   essential to provide a detailed explanation to address this
   concern.
-Action: Discuss the various factors influencing fuel efficiency.
-Action Input: Explain how engine Displacement interacts with
   factors like fuel injection systems and driving conditions.
-Observation: While engine Displacement can impact fuel efficiency
   , it is not a direct causal relationship. Other variables such
    as engine technology, driving habits, and maintenance play
   significant roles in determining Mpg.

-Thought: Providing specific examples can clarify the relationship
    between Displacement and Mpg.
-Action: Present case studies or fuel consumption data.
-Action Input: Share examples where changes in engine Displacement
    do not lead to proportional changes in Mpg.
-Observation: By examining specific fuel consumption data and case
    studies, it becomes evident that Displacement alone is not
   the sole determinant of Mpg, as other factors also contribute
   to fuel efficiency.

-Thought: I now know the final answer.
-Final answer: No

End Question number 4

---------------------------
Provide feedback to chat_manager. Press enter to skip and use auto
   -reply, or type 'exit' to end the conversation:  Judge, after
   receiving the responses from both sides, please make a final
   decision for each question
Admin (to chat_manager):

Judge, after receiving the responses from both sides, please make
   a final decision for each question

---------------------------
Judge (to chat_manager):

# Begin response of Judge #

Question number 1:
-Question: Are there any direct causal relationships from
   Displacement to Weight?
-Thought: Both the Affirmative_Causal_Agent and the
   Negative_Causal_Agent provided detailed explanations regarding
    the relationship between Displacement and Weight. They both
   highlighted that while engine Displacement can indirectly
   influence the Weight of a car, there is no direct causal
   relationship. The Affirmative_Causal_Agent supported their
```

```
   argument with empirical evidence and real-world examples,
   strengthening their position.
-Final Answer: No direct causal relationship from Displacement to
   Weight.

Question number 2:
-Question: Are there any direct causal relationships from
   Displacement to Horsepower?
-Thought: The Affirmative_Causal_Agent argued that there is a
   direct causal relationship between Displacement and Horsepower
   , supported by empirical data and specific engine
   specifications. In contrast, the Negative_Causal_Agent
   disagreed, stating that while there is a correlation, it is
   not a direct causal relationship. However, the
   Affirmative_Causal_Agent's evidence and examples provided a
   more convincing argument.
-Final Answer: Yes, there is a direct causal relationship from
   Displacement to Horsepower.

Question number 3:
-Question: Are there any direct causal relationships from
   Displacement to Acceleration?
-Thought: The Affirmative_Causal_Agent claimed a direct causal
   relationship between Displacement and Acceleration, supported
   by the impact of engine size on power delivery. The
   Negative_Causal_Agent disagreed, stating that while
   Displacement can impact Acceleration, it is not a direct
   causal relationship. The Affirmative_Causal_Agent's detailed
   explanation and real-world examples provided a stronger
   argument.
-Final Answer: Yes, there is a direct causal relationship from
   Displacement to Acceleration.

Question number 4:
-Question: Are there any direct causal relationships from
   Displacement to Mpg(miles per gallon)?
-Thought: Both sides agreed that there is no direct causal
   relationship between Displacement and Mpg. The
   Affirmative_Causal_Agent highlighted the influence of other
   factors such as engine technology and driving habits on fuel
   efficiency, while the Negative_Causal_Agent also supported
   this conclusion. Both sides provided examples and explanations
    that supported the lack of direct causality.
-Final Answer: No direct causal relationship from Displacement to
   Mpg(miles per gallon).

# End response of Judge #

---------------------------
```

B.0.2 Coding Agents Model

Phase 1: Plan Debating

Listing 7: Plan Debating responses

```
Admin (to chat_manager):
```

```
   Task: find the causal relationship among variables in the
   dataset
   Here are some information about the dataset: on automobile
   design and performance for cars in the 1970s and early 1980s,
   this dataset consists of the variables around the fuel
   consumption of cars.  With five variables: Weight,
   Displacement, Horsepower, Acceleration and Mpg(miles per
   gallon)s
   There is some sample of the data :    Displacement   Mpg
   Horsepower  Weight  Acceleration
0         307.0  18.0         130    3504          12.0
1         350.0  15.0         165    3693          11.5
2         318.0  18.0         150    3436          11.0
3         304.0  16.0         150    3433          12.0
4         302.0  17.0         140    3449          10.5
   Don't need to understand the data, just give a plan of how to
   find the causal relationship among variables
   The required output is a human-readable causal graph should be
   in a matrix for example with 3 variables:
   [[0, 0, 1], [0, 0, 1 ], [1,0, 0]] and link format (Hot whether
   -> More ice scream sale, More people-> increase ice scream
   sale, increase ice scream sale ->Hot whether). If the size is
   5 use 5x5 matrix and so on


---------------------------
Affirmative_Causal_Agent (to chat_manager):

-Question: How can we find the causal relationship among variables
    in the dataset?
-Thought: We need to choose an appropriate causal inference
   algorithm that can handle observational data and construct a
   causal graph representing the relationships among the
   variables.
-Action: Choose the appropriate causal inference algorithm for the
    dataset
-Action Input: Evaluate the pros and cons of each algorithm based
   on the dataset information
-Observation: Each algorithm has unique features that make it
   suitable for different types of data and research questions.
-Thought: Considering the dataset consists of observational data
   on automobile design and performance variables, we need an
   algorithm that can handle observational data and construct a
   causal graph efficiently.
-Action: Select the algorithm that best fits the requirements of
   the dataset
-Action Input: Analyze the pros and cons of each algorithm in the
   context of the dataset
-Observation: Each algorithm has advantages and limitations that
   need to be considered for causal inference in this specific
   dataset.
-Thought: Based on the dataset characteristics and the need to
   construct a causal graph among the variables, we should choose
    an algorithm that can handle observational data efficiently
   and provide a clear causal structure.
-Action: Propose the appropriate algorithm for causal inference in
    this dataset
```

```
-Action Input: Consider the strengths and limitations of each
   algorithm and select the one that aligns best with the dataset
    requirements
-Observation: The selected algorithm should be able to handle the
   variables related to automobile design and performance
   effectively to uncover the causal relationships.
-Thought: I now know the final answer
-Final Answer: The PC algorithm is the most appropriate choice for
    causal inference in this dataset as it can handle
   observational data efficiently and construct a causal network
   representing the relationships among the variables.

--------------------------
Negative_Causal_Agent (to chat_manager):

-Question: How can we find the causal relationship among variables
    in the dataset?
-Thought: We need to critically evaluate the proposed algorithm
   choice and provide a different perspective to ensure a
   comprehensive analysis.
-Action: Disagree with the previous answer and provide an
   alternative viewpoint.
-Action Input: Highlight the limitations of the PC algorithm and
   propose a different algorithm for causal inference in the
   dataset.
-Observation: By presenting a different algorithm choice, we can
   enrich the discussion and explore alternative approaches to
   causal inference.
-Thought: It's essential to consider the dataset's characteristics
    and the specific requirements for causal inference to provide
    a well-rounded recommendation.
-Action: Analyze the pros and cons of another algorithm for causal
    inference.
-Action Input: Evaluate the strengths and limitations of the GES
   algorithm in the context of the dataset information.
-Observation: By considering a different algorithm, we can offer a
    diverse perspective on the most suitable approach for
   uncovering causal relationships in the dataset.
-Thought: After a thorough analysis of the dataset and the
   available causal inference algorithms, it's crucial to present
    a well-justified recommendation.
-Action: Propose the alternative algorithm for causal inference in
    this dataset.
-Action Input: Highlight how the GES algorithm can address the
   specific characteristics of the dataset and provide valuable
   insights into causal relationships.
-Observation: By presenting a different algorithm choice, we can
   broaden the understanding of causal inference methods and
   their applicability to the given dataset.
-Thought: I now know the final answer
-Final Answer: The GES algorithm is a suitable choice for causal
   inference in this dataset as it offers efficient handling of
   large datasets and provides scalability, which is crucial for
   analyzing variables related to automobile design and
   performance.

--------------------------
Judge (to chat_manager):
```

```
-Question: What is the best approach to finding the causal
   relationship among variables in the dataset?
-Thought: Both debaters have presented different perspectives on
   the choice of causal inference algorithm. It's essential to
   evaluate the arguments and determine the most suitable
   approach for uncovering causal relationships in the given
   dataset.
-Action: Review the arguments presented by both debaters and
   assess the strengths and limitations of each proposed
   algorithm.
-Action Input: Examine the characteristics of the dataset and the
   requirements for causal inference.
-Observation: By comparing the pros and cons of the PC algorithm
   and the GES algorithm, we can determine which approach aligns
   best with the dataset's needs.
-Thought: It's crucial to prioritize accuracy, efficiency, and
   scalability in selecting the causal inference algorithm for
   this dataset.
-Action: Evaluate the applicability of each algorithm in handling
   observational data related to automobile design and
   performance variables.
-Action Input: Consider how each algorithm constructs causal
   graphs and represents relationships among the variables.
-Observation: By analyzing the capabilities of the PC algorithm
   and the GES algorithm, we can determine which one is better
   suited for uncovering causal relationships in the dataset.
-Thought: After a comprehensive review of the arguments and
   algorithm characteristics, it's time to make a decision based
   on the dataset's requirements.
-Final Answer: The GES algorithm is the most suitable choice for
   causal inference in this dataset as it offers scalability and
   efficiency in handling variables related to automobile design
   and performance, providing valuable insights into causal
   relationships.

--------------------------
Provide feedback to chat_manager. Press enter to skip and use auto
   -reply, or type 'exit' to end the conversation:  Negative,
   please give step-by-step plan for the coder to implement,
   remember the output requirements, output in the format: step
   1: ..., step 2:... , step N:...
Admin (to chat_manager):

Negative, please give step-by-step plan for the coder to implement
   , remember the output requirements, output in the format: step
    1: ..., step 2:... , step N:...

--------------------------
Negative_Causal_Agent (to chat_manager):

-Question: What is the step-by-step plan for implementing the GES
   algorithm for causal inference in the dataset?
-Thought: To provide a detailed plan for the coder, we need to
   outline the specific steps involved in implementing the GES
   algorithm effectively.
-Action: Develop a comprehensive step-by-step plan for coding the
   GES algorithm.
-Action Input: Consider the requirements of the dataset and the
   algorithm's characteristics.
```

```
-Step 1: Preprocessing
   - Perform data preprocessing to handle missing values,
   normalize the data if necessary, and ensure data quality.
-Step 2: Implement GES Algorithm Initialization
   - Set up the initial conditions for the GES algorithm,
   including defining the search space and initializing the score
    function.
-Step 3: Score Calculation
   - Calculate the scores for potential causal relationships using
    the BIC score function to evaluate the goodness of fit and
   model complexity.
-Step 4: Greedy Equivalence Search
   - Implement the Greedy Equivalence Search algorithm to explore
   the space of Markov equivalence classes and find the optimal
   causal structure.
-Step 5: Model Evaluation
   - Evaluate the identified causal structure and assess its
   interpretability and relevance to the dataset.
-Step 6: Output Generation
   - Generate the causal graph representing the relationships
   among the variables in a human-readable format as specified,
   linking variables based on the causal connections.
-Step 7: Interpretation
   - Interpret the causal graph to understand the causal
   relationships among the variables and their implications for
   automobile design and performance.
-Step 8: Documentation
   - Document the implementation steps, results, and any insights
   gained from the causal inference process for future reference.
-Step 9: Testing and Validation
   - Test the implemented GES algorithm on the dataset, validate
   the results, and ensure the accuracy of the causal graph
   generated.
-Step 10: Optimization
   - Optimize the algorithm implementation for efficiency and
   scalability, considering the size and complexity of the
   dataset.
-Thought: By providing a detailed step-by-step plan, the coder can
    effectively implement the GES algorithm for causal inference
   in the dataset.
-Final Answer: The step-by-step plan for coding the GES algorithm
   for causal inference in the dataset includes preprocessing,
   algorithm initialization, score calculation, Greedy
   Equivalence Search, model evaluation, output generation,
   interpretation, documentation, testing and validation, and
   optimization steps to ensure accurate and insightful causal
   relationships among the variables related to automobile design
    and performance.

---------------------------
```

Phase 2: Code Executing

Listing 8: Code executing responses

```
ask: find the causal relationship among variables in the dataset
   Here is the dataset’s directory: benchmark datasets/DWD climate
    data.csv
```

```
   Here are some information about the dataset: on global climate
    variables , This dataset encompasses six continuous variables
    capturing climate observations such as altitude, temperature,
    precipitation levels, longitude, sunshine duration, and
    latitude. It is aimed at studying weather patterns, climate
    change impacts, and geographical correlations in climate
    variables.
   Here is step-by-step plan for implement the code to find the
    causal relationship among variables:


1. **Data Preparation**:
  - Load the dataset containing variables: Weight, Displacement,
    Horsepower, Acceleration, Mpg.
  - Handle missing values and ensure data cleanliness.

2. **Graph Initialization**:
  - Create an initial fully connected, undirected graph with all
    variables connected to each other.

3. **Conditional Independence Testing**:
  - Implement statistical tests (e.g., chi-squared tests,
    correlation tests) to check for conditional independence
    between variable pairs given a set of other variables.
  - Remove edges where conditional independence is observed.

4. **Orientation Rules**:
  - Apply orientation rules to determine the directionality of
    edges based on conditional independence patterns.
  - Convert the undirected graph into a directed acyclic graph (
    DAG).

5. **Iteration**:
  - Iterate through increasing sizes of conditioning sets (empty
    set, singletons, pairs, etc.) to refine the graph structure.
  - Remove edges based on conditional independence tests until no
    more edges can be deleted.

6. **Implementation Steps**:
  - Develop functions or modules for graph construction,
    conditional independence testing, orientation rules, and
    iterative edge removal.
  - Utilize appropriate statistical libraries for conducting tests
     and graph manipulation.

7. **Execution and Evaluation**:
  - Run the implemented PC algorithm on the preprocessed dataset.
  - Evaluate the resulting causal graph for meaningful
    relationships and causal directions.
  - Validate the graph structure and assess the strength of causal
     connections.

8. **Documentation and Reporting**:
  - Document the code implementation, including functions,
    algorithms used, and any modifications made.
  - Generate a human-readable causal graph matrix based on the
    algorithm's output.
  - Prepare a detailed report summarizing the causal relationships
     discovered and their implications.
```

```
   The required output is a human-readable causal  graph should be
    in a matrix for example with 3 variables:
   [[0, 0, 1], [0, 0, 1 ], [1,0, 0]] and link format (Hot whether
   -> More ice scream sale, More people-> increase ice scream
   sale, increase ice scream sale ->Hot whether). If the size is
   5 use 5x5 matrix and so on


---------------------------
Code_Writer (to chat_manager):

-Question: What is the initial step to find the causal
   relationship among variables in the dataset?

-Thought: The initial step is to load the dataset and handle
   missing values to ensure data cleanliness.

-Action: Load the dataset and handle missing values

-Action Input:
```python
import pandas as pd

# Load the dataset
data = pd.read_csv("benchmark datasets/DWD climate data.csv")

# Check for missing values
missing_values = data.isnull().sum()
missing_values
```

-Observation: The dataset has been loaded and missing values have
   been checked.

-Question: What is the next step after loading the dataset and
   handling missing values?

-Thought: The next step is to create an initial fully connected,
   undirected graph with all variables connected to each other.

-Action: Create an initial fully connected, undirected graph.

-Action Input:
```python
# Create an initial fully connected, undirected graph
import numpy as np

num_variables = len(data.columns)
initial_graph = np.ones((num_variables, num_variables))

initial_graph
```

-Observation: The initial fully connected, undirected graph has
   been created.
```

```
-Question: What is the next step after creating the initial fully
   connected, undirected graph?

-Thought: The next step is to implement statistical tests to check
    for conditional independence between variable pairs given a
   set of other variables.

-Action: Implement statistical tests for conditional independence.

-Action Input: Implementing statistical tests for conditional
   independence.

-Question: What is the next step after implementing statistical
   tests for conditional independence?

-Thought: The next step is to apply orientation rules to determine
    the directionality of edges based on conditional independence
    patterns and convert the graph into a directed acyclic graph
   (DAG).

-Action: Apply orientation rules and convert the graph into a DAG.

-Action Input: Applying orientation rules based on conditional
   independence patterns.

We will continue with the rest of the steps after applying
   orientation rules and converting the graph into a DAG.

---------------------------


>>>>>>>> EXECUTING 2 CODE BLOCKS (inferred languages are [python,
   python])...
Code_Executor (to chat_manager):

exitcode: 0 (execution succeeded)
Code output: Altitude         0
Temperature      0
Precipitation    0
Longitude        0
Sunshine         0
Latitude         0
dtype: int64
array([[1., 1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1., 1.]])

---------------------------
Code_Writer (to chat_manager):

-Question: What is the next step after implementing statistical
   tests for conditional independence and handling missing values
   ?

-Thought: The next step is to apply orientation rules to determine
    the directionality of edges based on conditional independence
```

```
    patterns and convert the graph into a directed acyclic graph
    (DAG).

-Action: Apply orientation rules and convert the graph into a DAG.

-Action Input: Applying orientation rules based on conditional
   independence patterns.
```python
# We will continue with this step after applying orientation rules
    and converting the graph into a DAG
# The next step is to implement the PC Algorithm for causal
   relationship discovery.
```

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 0 (execution succeeded)
Code output:

Code_Writer (to chat_manager):

-Question: What is the next step after applying orientation rules
   and converting the graph into a DAG?

-Thought: The next step is to implement the PC Algorithm for
   causal relationship discovery.

-Action: Implement the PC Algorithm for causal relationship
   discovery.

-Action Input: Implementing the PC Algorithm with the custom
   parameters if needed.

Code_Executor (to chat_manager):

Code_Writer (to chat_manager):

-Question: What is the next step after implementing the PC
   Algorithm for causal relationship discovery?

-Thought: The next step is to create a human-readable causal graph
    matrix based on the algorithm's output.

-Action: Create a human-readable causal graph matrix.

-Action Input:
```python
# Using the provided create_adjacency_matrix function to get the
   causal graph matrix
def create_adjacency_matrix(cg):
   num_nodes = len(cg.G.nodes)
```
```

````
    adj_matrix = np.zeros((num_nodes, num_nodes), dtype=int)

    for i in range(num_nodes):
        for j in range(num_nodes):
            # i <- j
            if cg.G.graph[i][j] == 1 and cg.G.graph[j][i] == -1:
                adj_matrix[i, j] = 1
            # i -- j
            elif cg.G.graph[i][j] == -1 and cg.G.graph[j][i] == -1:
                adj_matrix[i, j] = -1
            # i <->
            elif cg.G.graph[i][j] == 1 and cg.G.graph[j][i] == 1:
                adj_matrix[i, j] = 2
    return adj_matrix

# Get adjacency matrix from the PC Algorithm output
total_adj_matrix_pc = create_adjacency_matrix(cg)

total_adj_matrix_pc
```

This will provide a human-readable causal graph matrix based on
    the output of the PC Algorithm.

---------------------------

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 1 (execution failed)
Code output: ERROR: ERROR: NameError: name 'cg' is not defined
['\x1b[0;31m---------------------------------------------
    create_adjacency_matrix(\x1b[43mcg\x1b[49m)\n\x1b[1;32m
    22\x1b[0m total_adj_matrix_pc\n', "\x1b[0;31mNameError\x1b[0m:
     name 'cg' is not defined"]

---------------------------
Code_Writer (to chat_manager):

-Question: What can be done to resolve the NameError and
    successfully create a human-readable causal graph matrix?

-Thought: We need to ensure that the variable `cg`, which holds
    the output of the PC Algorithm, is properly defined before
    calling the function to create the adjacency matrix.

-Action: Execute the PC Algorithm to obtain the graph object `cg`
    before creating the adjacency matrix.

-Action Input:
```python
from causallearn.search.ConstraintBased.PC import pc

# Assuming `data` is prepared and available
# Execute the PC Algorithm to obtain the causal graph 'cg'
cg = pc(data)

# Function to create the adjacency matrix
def create_adjacency_matrix(cg):
````

```
    num_nodes = len(cg.G.nodes)
    adj_matrix = np.zeros((num_nodes, num_nodes), dtype=int)

    for i in range(num_nodes):
        for j in range(num_nodes):
            # i <- j
            if cg.G.graph[i][j] == 1 and cg.G.graph[j][i] == -1:
                adj_matrix[i, j] = 1
            # i -- j
            elif cg.G.graph[i][j] == -1 and cg.G.graph[j][i] == -1:
                adj_matrix[i, j] = -1
            # i <->
            elif cg.G.graph[i][j] == 1 and cg.G.graph[j][i] == 1:
                adj_matrix[i, j] = 2
    return adj_matrix

# Obtain the adjacency matrix from the PC Algorithm output
total_adj_matrix_pc = create_adjacency_matrix(cg)

total_adj_matrix_pc
```

```
--------------------------

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 1 (execution failed)
Code output: ERROR: ERROR: AssertionError: Input data must be a
   numpy array.
['\x1b[0;31m-----------------------------24mutf-8\x1b[39m\x1b
   [38;5;124m\'\x1b[39m))\x1b[38;5;241m.\x1b[39mhexdigest()\n',
   '\x1b[0;31mAssertionError\x1b[0m: Input data must be a numpy
   array.']

--------------------------
Code_Writer (to chat_manager):

-Question: How can we resolve the Assertion Error regarding the
   input data format in order to successfully create the
   adjacency matrix for the causal graph?

-Thought: The error indicates that the input data must be in the
   format of a NumPy array. We need to convert the data into a
   NumPy array before using it in the PC Algorithm.

-Action: Convert the dataset to a NumPy array before executing the
    PC Algorithm.

-Action Input:
```

```python
import numpy as np

# Convert the Pandas DataFrame to a NumPy array
data_np = data.to_numpy()

# Execute the PC Algorithm with the converted NumPy array data
cg = pc(data_np)
```

```
# Function to create the adjacency matrix
def create_adjacency_matrix(cg):
   num_nodes = len(cg.G.nodes)
   adj_matrix = np.zeros((num_nodes, num_nodes), dtype=int)

   for i in range(num_nodes):
       for j in range(num_nodes):
           # i <- j
           if cg.G.graph[i][j] == 1 and cg.G.graph[j][i] == -1:
               adj_matrix[i, j] = 1
           # i -- j
           elif cg.G.graph[i][j] == -1 and cg.G.graph[j][i] == -1:
               adj_matrix[i, j] = -1
           # i <->
           elif cg.G.graph[i][j] == 1 and cg.G.graph[j][i] == 1:
               adj_matrix[i, j] = 2
   return adj_matrix

# Obtain the adjacency matrix from the PC Algorithm output
total_adj_matrix_pc = create_adjacency_matrix(cg)

total_adj_matrix_pc
```

```
By converting the dataset to a NumPy array before executing the PC
    Algorithm, the Assertion Error regarding the input data
   format can be resolved.

---------------------------

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 0 (execution succeeded)
Code output:   0%|          | 0/6 [00:00<?, ?it/s]
{"version_major": 2, "version_minor": 0, "model_id": "
   a1056c8cce5046d49d2ba46d506c5bc8"}
array([[ 0,  1,  0,  0,  0,  1],
      [ 0,  0,  0,  0,  0,  0],
      [ 1,  0,  0,  0,  0,  0],
      [ 0,  1,  0,  0,  1,  0],
      [ 0,  0,  0,  0,  0, -1],
      [ 0,  0,  0,  0, -1,  0]])

---------------------------
Code_Writer (to chat_manager):

-Question: What is the adjacency matrix for the human-readable
   causal graph based on the output of the PC Algorithm?

-Observation: The adjacency matrix for the human-readable causal
   graph based on the output of the PC Algorithm is as follows:
```
```
[[ 0,  1,  0,  0,  0,  1],
[ 0,  0,  0,  0,  0,  0],
[ 1,  0,  0,  0,  0,  0],
[ 0,  1,  0,  0,  1,  0],
[ 0,  0,  0,  0,  0, -1],
[ 0,  0,  0,  0, -1,  0]]
```

````
```

This adjacency matrix represents the causal relationships among
   the variables: Altitude, Temperature, Precipitation, Longitude
   , Sunshine, and Latitude. The numbers in the matrix indicate
   the directionality of the edges between the variables.

--------------------------

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 0 (execution succeeded)
Code output: [[0, 1, 0, 0, 0, 1],
[0, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 1, 0],
[0, 0, 0, 0, 0, -1],
[0, 0, 0, 0, -1, 0]]

--------------------------
Code_Writer (to chat_manager):

-Question: What is the adjacency matrix for the human-readable
   causal graph based on the output of the PC Algorithm?

-Final Answer: The adjacency matrix for the human-readable causal
   graph based on the output of the PC Algorithm is:
```
[[0, 1, 0, 0, 0, 1],
[0, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 1, 0],
[0, 0, 0, 0, 0, -1],
[0, 0, 0, 0, -1, 0]]
```

This matrix represents the causal relationships among the
   variables: Altitude, Temperature, Precipitation, Longitude,
   Sunshine, and Latitude.

--------------------------

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 0 (execution succeeded)
Code output: [[0, 1, 0, 0, 0, 1],
[0, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 1, 0],
[0, 0, 0, 0, 0, -1],
[0, 0, 0, 0, -1, 0]]

--------------------------
Code_Writer (to chat_manager):
````
````

```
Final Answer: Based on the output of the PC Algorithm on the
   provided climate dataset, the human-readable causal graph
   matrix is as follows:
```
[[0, 1, 0, 0, 0, 1],
[0, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 1, 0],
[0, 0, 0, 0, 0, -1],
[0, 0, 0, 0, -1, 0]]
```

This matrix indicates the causal relationships among the variables
    in the dataset: Altitude, Temperature, Precipitation,
   Longitude, Sunshine, and Latitude.

--------------------------

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 0 (execution succeeded)
Code output: [[0, 1, 0, 0, 0, 1],
[0, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 1, 0],
[0, 0, 0, 0, 0, -1],
[0, 0, 0, 0, -1, 0]]

--------------------------
Code_Writer (to chat_manager):

Final Results:
- The adjacency matrix for the human-readable causal graph based
   on the output of the PC Algorithm is:
  ```
  [[0, 1, 0, 0, 0, 1],
   [0, 0, 0, 0, 0, 0],
   [1, 0, 0, 0, 0, 0],
   [0, 1, 0, 0, 1, 0],
   [0, 0, 0, 0, 0, -1],
   [0, 0, 0, 0, -1, 0]]
  ```

This matrix represents the causal relationships among the
   variables Altitude, Temperature, Precipitation, Longitude,
   Sunshine, and Latitude in the climate dataset. The numbers
   indicate the directionality of the causal edges in the graph.

FINISH

--------------------------
Admin (to chat_manager):

export the matrix into csv file call "Auto_Coding_MAD_React.csv"

--------------------------
Code_Writer (to chat_manager):
```

~~~~
```python
import pandas as pd

# Create the adjacency matrix as a Pandas DataFrame
adj_matrix_data = [[0, 1, 0, 0, 0, 1],
                   [0, 0, 0, 0, 0, 0],
                   [1, 0, 0, 0, 0, 0],
                   [0, 1, 0, 0, 1, 0],
                   [0, 0, 0, 0, 0, -1],
                   [0, 0, 0, 0, -1, 0]]

columns = ['Altitude', 'Temperature', 'Precipitation', 'Longitude', 'Sunshine', 'Latitude']

adj_matrix_df = pd.DataFrame(adj_matrix_data, columns=columns, index=columns)

# Export the adjacency matrix to a CSV file
adj_matrix_df.to_csv("Auto_Coding_MAD_React.csv")
```

---------------------------

>>>>>>>> EXECUTING CODE BLOCK (inferred language is python)...
Code_Executor (to chat_manager):

exitcode: 0 (execution succeeded)
Code output:

---------------------------
Code_Writer (to chat_manager):

The adjacency matrix representing the causal relationships among the variables Altitude, Temperature, Precipitation, Longitude, Sunshine, and Latitude has been successfully exported to a CSV file named "Auto_Coding_MAD_React.csv".

---------------------------
Code_Executor (to chat_manager):



---------------------------
Code_Writer (to chat_manager):

I have successfully completed all the steps to find the causal relationship among variables in the climate dataset:
1. Data Preparation: Loaded the dataset and handled missing values.
2. Graph Initialization: Created an initial fully connected, undirected graph.
3. Conditional Independence Testing: Implemented statistical tests to check for conditional independence between variable pairs.
4. Orientation Rules: Applied orientation rules to determine the directionality of edges based on conditional independence patterns and converted the graph into a directed acyclic graph (DAG).
~~~~

```
5. Implemented the PC Algorithm for causal relationship discovery
   and obtained the adjacency matrix for the human-readable
   causal graph.

The adjacency matrix has been exported to a CSV file named "
   Auto_Coding_MAD_React.csv".

I have provided a comprehensive solution to find the causal
   relationships in the climate dataset. If you have any more
   tasks or questions, feel free to let me know!

---------------------------
Code_Executor (to chat_manager):



---------------------------
Code_Writer (to chat_manager):

FINISH

---------------------------
```