SEA: Stateful Execution Environment for Conversational Big Data Analytics

Anonymous Author(s)
Affiliation
Address
email

Abstract

Applying large language model (LLM) agents to conversational data analytics is challenging, as existing agents often operate statelessly, leading to inefficiency and a fragmented user experience in multi-turn interactions. We argue that the agent's environment should explicitly encode the domain's predictable workflow. This reframes the agent's role from complex, open-ended planning to a more tractable task: strategically selecting where to resume a structured process to maximize state reuse. To this end, we introduce the **Stateful Execution Environment (SEA)**, a framework that represents the data analysis workflow as a Directed Acyclic Graph (DAG). A key feature of SEA is its **dual-representation state model**, which decouples a lightweight, symbolic state graph for the LLM planner from a full computational state graph used for execution. We evaluate SEA on GloboMart, a new large-scale benchmark for conversational data analytics. Our experiments show that the planner achieves over **95%** accuracy on its reframed task, leading to an **84%** end-to-end task success rate and a **36%** reduction in average latency on stateful follow-up queries. Our work demonstrates that designing environments with strong workflow priors is a critical step toward building more efficient and reliable agents for domain-specific reasoning.

1 Introduction

Large language models (LLMs) have recently enabled a new class of intelligent agents that can interact with software tools, data, and users through natural language Wang et al. [2025], Schick et al. [2023]. A central insight from this line of work is that the design of the *environment*—the context in which the agent perceives state, chooses actions, and receives feedback—plays a critical role in determining the agent's efficiency and reliability. While much effort has gone into scaling models and datasets, the question of how to design environments for domain-specific reasoning remains open.

One such domain where this question is particularly important is **conversational big data analytics**. Analysts often want to explore large and complex datasets using natural language, without needing to know database schemas or programming details. This goal has motivated work in natural language to SQL translation Wang et al. [2019], Lei et al. [2024], Liu et al. [2025], visualization generation Dibia [2023], and interactive data exploration systems Ding et al. [2023], Weng et al. [2024]. These systems show strong results on individual tasks. However, the end-to-end workflow remains fragmented: finding the right tables, generating complex joins, performing localized analysis, and communicating insights are usually handled by separate tools, with no open-source unified framework that ties them together in a cohesive process.

Our work starts from a simple but important observation: the workflow of data analysis is not arbitrary. Unlike open-ended tool-using agents, where the sequence of actions must be discovered Yang et al. [2023], data analysis follows a *predictable sequence of stages*: data discovery, data subsetting,

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

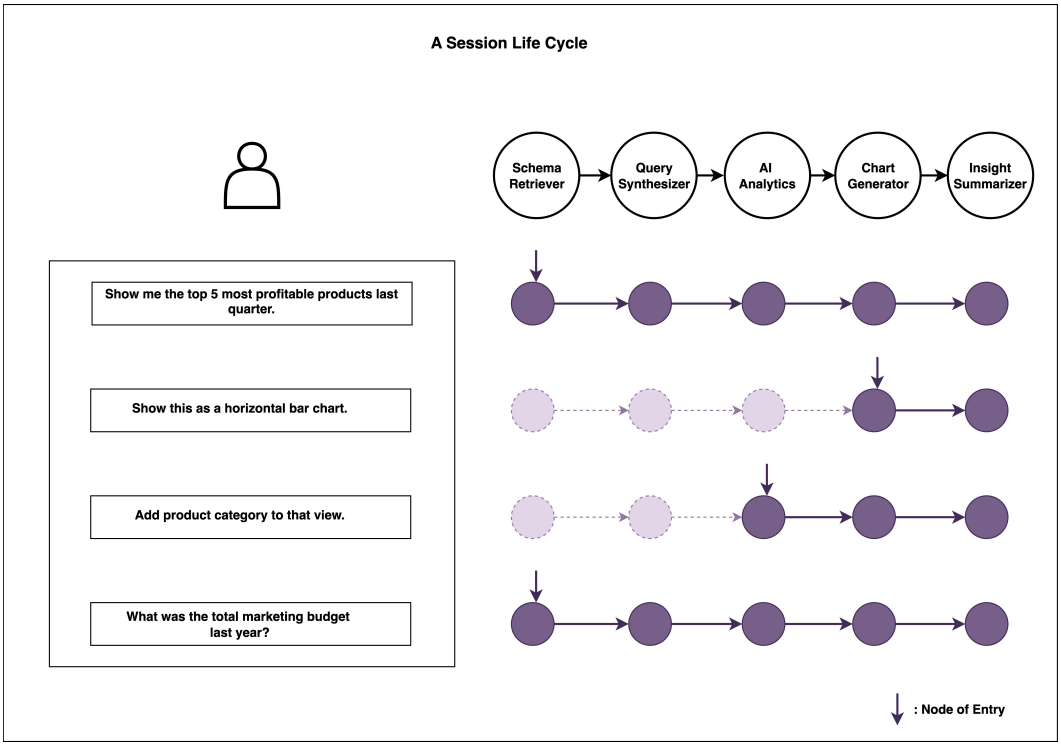

Figure 1: A high-level overview of the Stateful Execution Environment (SEA). The agent interacts with a structured environment representing the data analysis workflow as a DAG. For follow-up queries, the Planner's primary role is to select the optimal entry-point node (e.g., AIAnalytics) to resume execution, maximizing state reuse from the cached Main State DAG and avoiding redundant, high-latency operations like data subsetting.

localized analysis, and finally presentation of insights. We argue that by explicitly encoding this workflow into the environment, we can shift the agent's role from complex workflow planning to the more tractable task of *strategic optimization within a structured process*. This is the central thesis of our work. Essenstially, for follow-up queries, the planner's prime objective becomes a *node classification* problem: identifying the optimal point in the workflow to resume from, in order to maximize state reuse and ensure a fast, efficient conversation.

To this end, we introduce the **Stateful Execution Environment (SEA)**, a framework for conversational big data analytics built around this principle (see Figure 1 for a high-level overview). SEA represents the analytical workflow as a directed acyclic graph (DAG), where each node corresponds to a specialized tool (for discovery, SQL-based subsetting, Python-based analysis, and visualization) and the directed edges represents the immutable data dependencies between them. This structure deliberately separates the compute-heavy, high-latency stage of Data Subsetting from the fast, localized stage of Insight Generation. To make this state observable to the agent, we introduce a **dual-representation state design**: a *Main State DAG* caches full computational results (e.g., large dataframes) for the tools to access within the action space, while a lightweight *Summary DAG* provides compressed metadata to the LLM-based planner. The planner produces structured execution plans, while a deterministic executor enforces workflow consistency and parallelizes tasks in the action space. To support natural conversations, SEA also introduces a session lifecycle with explicit resets, preventing context drift while aligning with how analysts typically work (deep dive on one topic, then switch).

To study SEA, we curated a realistic, large-scale evaluation setting. We constructed an e-commerce dataset **GloboMart** with 8 tables (up to 6M rows and upto 14 columns each), designed to capture the challenges of big data analytics and follows the standard setup of e-commerce big data. On top of this, we built a benchmark of 100 conversational queries that include topic switches, complex joins, and multi-turn analysis. The field is increasingly recognizing that evaluating an agent based solely on final task success is an insufficient metric Armony et al. [2025]. Thus, we evaluate the system at

multiple levels: (i) the Planner's accuracy on the node classification task, (ii) the correctness of data discovery and join generation, and (iii) the end-to-end quality of system responses judged by human evaluators. Our evaluation confirms the efficacy of this structured approach: the agent's planner achieves over 95% accuracy on its simplified task, while stateful execution leads to a 36% reduction in average latency for follow-up queries. The full dataset, benchmark, and code has been provided in the supplementary material. These will be also released to the community [1].

Our primary contributions are:

- We propose **SEA**, a structured environment that operationalizes the inherent workflow of data analytics, enabling more efficient and reliable agentic reasoning.
- We introduce a **dual-representation state DAG model**, balancing heavy computational state for execution with compressed symbolic state for planning.
- We provide a **large-scale dataset and conversational benchmark** tailored to big data analytics, supporting systematic evaluation of both agent-level planning and end-to-end system quality.
- We demonstrate through a multi-faceted evaluation that our structured environment enables an agent to achieve high accuracy on complex analytical dialogues.

By grounding agent design in the structure of the domain, our work highlights a general lesson: environments that encode workflow priors can lead to agentic systems that are not only more efficient, but also more transparent and reliable in practice.

2 Related Works

This work intersects with agentic AI for data analytics, planning architectures, and state management in conversational systems. While significant progress has been made, challenges remain in creating agents that can efficiently and reliably conduct iterative, multi-turn data exploration.

2.1 Agentic Architectures for Data Analytics

The application of LLMs to data analysis has evolved from direct code generation Chen et al. [2021] to agentic systems built on frameworks like LangChain Chase [2022]. While versatile, these general-purpose agents often operate reactively, re-evaluating context on each turn, which is inefficient for structured, state-dependent analytical workflows.

A prominent agent architecture is the **ReAct** framework Yao et al. [2023], which externalizes the agent's reasoning process via a Thought-Action-Observation loop. While this offers high transparency and is effective for dynamic tasks, its step-by-step deliberation incurs significant latency and token costs in more structured domains like data analysis.

To improve efficiency, the **Planner-Executor** model separates a high-level planner from a deterministic executor Wang et al. [2023]. In this model, the planner typically must discover the entire workflow from scratch. An alternative approach, which we explore, is to provide the agent with a predefined workflow structure, shifting the planning task from open-ended generation to strategic selection within that structure.

2.2 State Management as the Core Challenge

The ephemeral context of LLMs makes robust state management a central challenge for any task requiring continuity, as interruptions can reset an agent's progress Packer et al. [2023], Shinn et al. [2023]. Stateful design patterns have emerged to address this, such as graph-based workflows that pass state objects between nodes LangChain Team [2024], or Finite State Machines (FSMs) that provide process grounding through explicit state transitions Wu et al. [2024a].

A key distinction in this area is between *process state*—the current step in an execution flow—and *analytical state*, which is the evolving semantic context of an exploration (e.g., cumulative filters,

[1] Codes, Dataset, and Benchmark available at: `https://osf.io/buxma/files/osfstorage?view_only=f7de66430a7b42e0acbc6330ecedd255`

transformations). While existing solutions primarily address the former, the latter is often left unstructured within conversational history, presenting a challenge for efficient, long-term reasoning in analytical dialogues.

2.3 Core Capabilities for Data Interaction

An effective data agent must wield a suite of specialized tools, whose state-of-the-art is continually advancing.

Text-to-SQL. Accurate SQL generation over complex databases requires effective **schema linking** to provide the LLM with only the relevant schema subset Li et al. [2024], An et al. [2025]. This is commonly solved with retrieval-based methods Wu et al. [2024b], with advanced multi-hop techniques used to find relationships across disparate tables Zhang et al. [2024].

Data-to-Insight. Communicating results involves **Data-to-Text** generation, where ensuring *semantic fidelity* to the source data is a primary challenge Harkous et al. [2020], or **Text-to-Visualization**. The dominant technique for visualization is prompting an LLM to generate executable plotting code Dibia [2023], Weng et al. [2025]. The components in our proposed system build upon these state-of-the-art capabilities, integrating them into a managed, stateful environment.

3 The SEA Methodology

The Stateful Execution Environment (SEA) is a specialized infrastructure designed to facilitate efficient, multi-turn agentic reasoning for complex data analytics. Our methodology is founded on the principle that by explicitly modeling the domain's inherent, predictable workflow structure, we can create a highly optimized environment. We formalize this workflow as a Stateful Execution Directed Acyclic Graph (DAG), which serves as the central state representation for our agent. This allows us to decompose the overall process into two distinct macro-stages: (1) large-scale, high-latency **Data Subsetting**, followed by (2) localized, low-latency **Insight Generation**. This process is operationalized by a Planner–Executor architecture, as illustrated in Figure 2.

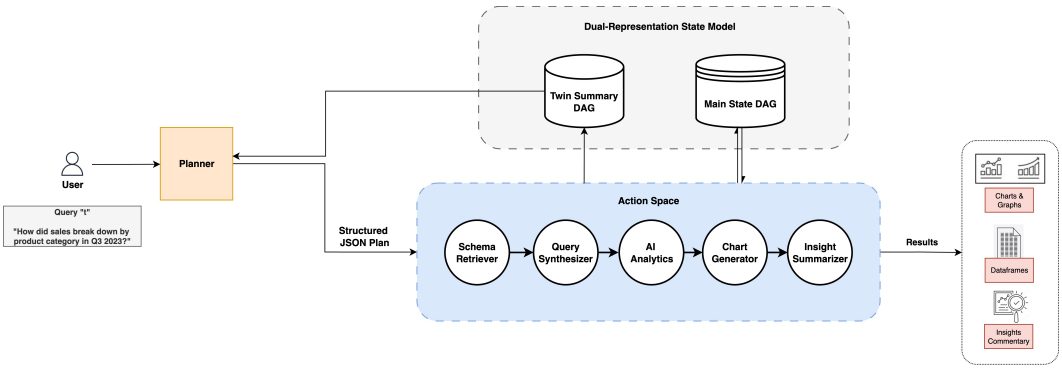

Figure 2: The SEA architecture. At turn t, the **Planner** receives user query q_t and the lightweight Twin Summary DAG S'_{t-1}. It performs node classification to select an entry-point v_{entry} and generates a plan P_t. The **Executor** then executes the plan, calling the necessary tools which access the heavy artifacts in the Main State DAG S_{t-1}. Each tool updates both state representations, producing the final state (S_t, S'_t) for the next turn.

Having defined the environment's structure, we now formalize the agent's interaction with it through its three core components: the action space ($\mathcal{A}$), the state space ($\mathcal{S}$), and the core operational loop governed by the Planner ($\mathcal{P}$) and Executor ($\mathcal{X}$).

3.1 The Action Space ($\mathcal{A}$): A Composable Analytics Toolset

The agent's capabilities are materialized through the action space $\mathcal{A}$, a suite of five specialized, deterministic tools. These tools are the vertices of our workflow graph and are designed for fine-grained, collaborative execution.

SchemaRetriever This tool initiates the workflow by mapping user intent to specific data tables. It operates a two-stage process: an offline semantic indexing step where enriched JSON table summaries are embedded into a vector database, and an online retrieve-and-rerank mechanism that uses dense retrieval followed by LLM-based reranking to return the precise set of tables required, critically enabling multi-table `JOIN` reasoning. The tool also returns a reasoning string to justify its chosen table names.

QuerySynthesizer This tool executes the Data Subsetting macro-stage. It receives the table schemas from the `SchemaRetriever` and the user's intent from the Planner. It then generates and executes a complex SQL query against the data warehouse, returning a single, unified pandas DataFrame. With the collaboration of `SchemaRetriever`, it is also able to handle multi-table `JOIN` cases.

AIAnalytics This tool begins the Localized Insight Generation macro-stage. It operates on the manageable DataFrame subset from the previous step, generating and executing pandas code to produce the final, user-facing analytical DataFrame.

InsightSummarizer A data-to-text agent that ingests the final analyzed DataFrame and generates a concise, narrative summary of the key findings tailored to the user intent.

ChartGenerator A code-generating agent that takes the final DataFrame's structured schema as input. It determines a suitable visualization type, then generates and executes Python code to render the chart.

3.2 The State Space ($\mathcal{S}$): A Dual-Representation DAG

The environment's state is encoded in a Stateful Execution Directed Acyclic Graph (DAG), $G = (V, E)$. For the conversational analytics task in our case, this is a 5-node linear graph: `SchemaRetriever` $\rightarrow$ `QuerySynthesizer` $\rightarrow$ `AIAnalytics` $\rightarrow$ {`InsightSummarizer`, `ChartGenerator`}. Vertices V correspond to the tools in our action space, and directed edges E represent the immutable data dependencies. To resolve the fundamental tension between the need for a rich, high-fidelity state and the context limitations of LLMs, we introduce a novel **Dual-Representation State Model**.

The Main State DAG (S) This is the ground-truth state of the environment. Each vertex $v \in V$ caches the complete, computationally "heavy" artifact from its last execution—the full pandas DataFrame, the rendered image file. This state is exclusively accessed by the deterministic Executor and the tools themselves.

The Twin Summary DAG (S') This is a lightweight, symbolic representation of S, designed explicitly as the *observation provided to the Planner Agent*. Each vertex in S' stores high-signal, low-token metadata that proxies the Main State: the chosen table names along with a reasoning string for this choice, generated SQL and pandas code, and structured JSON schemas of intermediate DataFrames. This allows the Planner to reason about the environment's state without being inundated by raw data.

3.3 The Core Operational Loop: Planner–Executor Interaction

Remark. When $v_{\text{entry}} = v_0$, the purge clears stale artifacts but does not invalidate P_t: the plan is specified relative to the root and depends only on (q_t, M) and the fixed DAG structure, not on previous heavy artifacts. The triggering query q_t also seeds the new session memory, ensuring the Planner never operates from an empty context.

Algorithm 1: SEA Planner–Executor Operational Loop

Input : Incoming stream of user queries $\{q_t\}_{t=1}^{T}$ (revealed sequentially)
Output : Per-turn analysis artifacts (summary + chart)

Initialize Main State DAG S_0, Twin Summary DAG S'_0, Planner memory $M \leftarrow \varnothing$
for *each incoming query* q_t **do**
  Update Planner memory $M \leftarrow M \cup \{q_t\}$
  `// -- Planner Phase --`
  $v_{\text{entry}} \leftarrow \pi_{\text{select}}(S'_{t-1}, q_t, M)$
  $P_t \leftarrow$ ordered JSON plan over subgraph $\mathcal{G}(v_{\text{entry}})$
  `// -- State Carry-Forward --`
  $S_t \leftarrow S_{t-1}; \quad S'_t \leftarrow S'_{t-1}$
  `// -- Executor Phase --`
  for *tool* $a \in P_t$ **do**
    Execute a deterministically with parameters
    $S_t[v_a] \leftarrow \text{artifact}(a)$
    $S'_t[v_a] \leftarrow \text{metadata}(a)$
  end
  `// -- Session Lifecycle Check --`
  if $v_{entry} = v_0$ `// root node, SchemaRetriever`
  then
    `// topic switch detected` $\rightarrow$ `start new session`
    Reset $S_t, S'_t \leftarrow \varnothing$
    Reset $M \leftarrow \{q_t\}$ `// memory primed with current query`
  end
end

The interaction between the agent and the environment is governed by this operational loop. The Planner ($\mathcal{P}$) is stochastic and exploratory; the Executor ($\mathcal{X}$) is deterministic and state-updating. This separation guarantees both flexibility in planning and reliability in execution.

The Planner Agent ($\mathcal{P}$) The Planner is an LLM-based agent whose primary task is reframed from workflow discovery to strategic path optimization. We model this as a **node classification** problem. At each conversational turn t, the Planner's policy, π, takes the user query q_t, the Twin Summary DAG S'_{t-1}, and the session memory M to select an optimal *entry-point* vertex, $v_{\text{entry}} \in V$:

$$v_{\text{entry}} = \pi_{\text{select}}(S'_{t-1}, q_t, M). \tag{1}$$

This vertex represents the earliest stage in the workflow whose state is invalidated by the new query. The Planner then generates an *ordered* JSON plan, P_t, for subgraph execution starting from v_{entry}, including contextually enriched parameters for each tool.

The Deterministic Executor ($\mathcal{X}$) The Executor functions as the state transition mechanism of the environment. It receives the plan from $\mathcal{P}$ and deterministically executes the specified tool sequence. For each tool call, it updates both the Main State DAG (S_t) with the heavy artifact and the Twin Summary DAG (S'_t) with its corresponding metadata. The Executor also manages control flow, such as the parallel execution of the `ChartGenerator` and `InsightSummarizer`.

3.4 Context and Memory: A Session-Lifecycle Approach

SEA employs a **session-based memory lifecycle**, a deliberate design choice that diverges from generic, infinite-context mechanisms. Generalist methods, such as retrieval over full conversational history, risk introducing context drift and semantic ambiguity when applied to the precise, state-dependent nature of data analysis, where remnants of a prior, unrelated analysis can degrade planning accuracy.

Our session-lifecycle approach, formalized in the final block of Algorithm 1, provides a robust solution. A session is defined as a continuous dialogue on a single analytical topic. The Planner's memory (M) within a session is limited to the user's raw query history, used for contextual enrichment.

When the Planner identifies a query as a complete topic switch—a decision materialized when it selects the initial node of the DAG as its entry-point ($v_{\text{entry}} = v_0$)—it triggers a **purge event**. The entire state, including both DAG representations (S and S') and the Planner's internal memory (M), is reset.

This design represents a principled trade-off. We sacrifice zero-shot state reuse for cross-topic comparisons, which must be handled as new sessions. In return, we gain a robust, self-cleaning memory system that entirely prevents context pollution and ensures planning reliability across long-running interactions, aligning with the observed "deep dive then switch" workflow of human data analysts.

Details on system implementation has been provided in Appendix A.

4 The GloboMart Benchmark Environment

To rigorously evaluate SEA's multi-turn, join-heavy reasoning capabilities, we developed the **GloboMart** benchmark, as existing datasets lack the necessary conversational depth and schematic complexity. The environment features a synthetic but realistic e-commerce data warehouse with 8 interconnected tables (up to 6M rows each) structured in a join-heavy **star schema**, an industry standard for analytics.

Complementing the data, we curated a benchmark of **100 conversational queries** structured as multi-turn dialogues. These are designed to probe key capabilities, including an agent's ability to handle complex multi-table joins, reuse state efficiently in follow-ups, and manage memory during abrupt topic switches. A detailed description of the dataset architecture and the benchmark are provided in Appendix B and released publicly with our source code.[2]

5 Evaluation

To assess the efficacy of the SEA framework, we conduct a comprehensive, multi-level evaluation on our new GloboMart benchmark. Due to the absence of publicly available, end-to-end conversational analytics systems that perform both data discovery and insight generation over large data warehouses, a direct comparative baseline is not feasible. Therefore, we adopt a rigorous intrinsic evaluation methodology, designed to answer three core questions: (1) Does the Planner Agent reason correctly by selecting the optimal execution path? (2) Are the system's key components and final output accurate and efficient? (3) What does the Planner's dynamic behavior look like in a real conversation?

5.1 Planner Performance: Entry-Point Classification and Path Fidelity

Our central thesis is that SEA reframes the Planner's task into a strategic node classification problem. We evaluate this directly by analyzing the execution plans generated by the Planner against manually annotated ground-truth plans for each of our 100 benchmark queries.

Entry-Point Accuracy We treat the selection of the first tool in the plan as a direct measure of the Planner's success at the node classification task. The accuracy is measured as the percentage of queries where the Planner's chosen entry-point action exactly matches the ground truth.

Path Fidelity To measure the overall alignment of the generated plan's reasoning trajectory with the ground truth, we compute the normalized Levenshtein distance. This metric captures the total misalignment by calculating the minimum number of single-token edits (insertions, deletions, or substitutions) required to change one sequence into the other. For two tool call sequences P_i and P_i^* of lengths m and n, the Levenshtein distance $L(P_i, P_i^*)$ is formally defined as:

$$L_{i,j} = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} L_{i-1,j} + 1 \\ L_{i,j-1} + 1 \\ L_{i-1,j-1} + \mathbb{I}(P_i[i] \neq P_i^*[j]) \end{cases} & \text{otherwise.} \end{cases} \tag{2}$$

[2]GloboMart dataset and the SEA source code available at: `https://osf.io/buxma/files/osfstorage?view_only=f7de66430a7b42e0acbc6330ecedd255`

We report the normalized distance, $L(P_i, P_i^*)/\max(m, n)$, to account for varying plan lengths.

5.2 Task Success and System Efficiency

We evaluate the accuracy of a critical component and the final system output using a strict, binary classification (pass/fail) approach, complemented by an analysis of conversational latency.

Schema Retrieval Accuracy The ability of the `SchemaRetriever` to identify the correct set of tables is a prerequisite for generating correct joins. We evaluate this as a binary task: a 'pass' is awarded only if the set of table names returned by the tool *exactly* matches the ground-truth set required to answer the query.

Final Output Correctness The end-to-end system response for each of the 100 queries was independently judged by two expert data analysts. A response is marked as a 'pass' only if both analysts agreed that the final artifacts (DataFrame, visualization, and commentary) were entirely correct and fulfilled the user's intent. Any error, including minor miscalculations or misleading commentary, resulted in a 'fail'.

Conversational Latency Beyond correctness, a key measure of success for a conversational system is its interactive performance. To demonstrate the practical efficiency gains from our stateful architecture, we present illustrative latency measurements for fresh queries versus stateful follow-up queries. This analysis quantifies the significant reduction in response time achieved by bypassing the expensive Data Subsetting stage on follow-up queries.

5.3 Qualitative Analysis of Planner Behavior

To provide an intuitive, qualitative understanding of the Planner's dynamic path selection in a multi-turn dialogue, we refer to the visualization presented in Figure 1. This figure renders the execution paths chosen by the Planner for a representative 4-query conversational slice from our benchmark (one initial query and three subsequent follow-ups). It offers a clear demonstration of the "multi-port entry" mechanism in action, showing how the Planner intelligently reuses state by initiating subsequent executions at deeper nodes in the DAG (as conceptualized in Figure 2), thereby avoiding redundant computation. The complete, annotated evaluation benchmark is provided in the supplementary material for full reproducibility.

6 Results and Discussion

Our evaluation demonstrates the effectiveness of the SEA framework across planner performance, component accuracy, and system efficiency. The key quantitative results on the GloboMart benchmark are summarized in Table 1.

Table 1: Quantitative evaluation results on the GloboMart benchmark. Planner performance is measured by its ability to select the correct entry-point and follow the optimal execution path. System accuracy is evaluated on the critical schema retrieval step and the end-to-end output. Latency figures highlight the efficiency gains from state reuse.

Category	Metric	Score
Planner Performance		
	Entry-Point Accuracy	95.65%
	Path Fidelity (1 - Norm. Levenshtein)	92.68%
System Accuracy & Component-Level Success		
	Schema Retrieval Accuracy	84.06%
	Final Output Correctness (Human Eval)	84.06%
System Efficiency		
	Avg. Latency (Fresh Query)	53.40 s
	Avg. Latency (Stateful Follow-up)	34.10 s

The results strongly support our central thesis. The Planner achieves **95.65% accuracy** in the entry-point classification task, indicating that it can reliably identify the optimal point to resume the workflow. High path fidelity (92.68%) further confirms that once the entry-point is chosen, the subsequent plan generation is robust. This validates our approach of simplifying the agent's task from open-ended planning to strategic node selection within a structured environment.

The end-to-end system achieves a final correctness of **84.06%** under strict human evaluation. Crucially, this is identical to the Schema Retrieval Accuracy. This alignment reveals that the primary bottleneck and source of failure is the initial data discovery step. When the `SchemaRetriever` correctly identifies the required tables, the downstream deterministic tools (`QuerySynthesizer`, `AIAnalytics`) are highly reliable in executing the correct logic.

The practical benefit of SEA's stateful design is evident in the latency metrics. Stateful follow-up queries bypass the high-latency Data Subsetting stage, where the `QuerySynthesizer`'s **server call to the big data backend consumes the most time**. As illustrated in the example in Figure 3, this state reuse can lead to dramatic performance gains. The follow-up query, which enters the DAG at a later stage, achieves a **70.7% reduction in response time**, which is critical for maintaining a fluid, interactive conversational experience.

Figure 3: A conversational example illustrating latency reduction via state reuse. **Query 1** initiates a new analysis, running the full DAG and incurring high latency from the data subsetting call. **Query 2**, a direct follow-up, reuses the cached DataFrame, entering the DAG at a later stage (`AIAnalytics`) and achieving a 70.7% reduction in response time.

Query	Latency
1. `What was the total marketing budget last year?`	59.96 s
2. `Compare budget to total spend by channel.`	17.57 s

7 Conclusion

In this work, we introduced SEA, a stateful execution environment designed to address the challenges of efficiency and reliability in conversational big data analytics. By modeling the analytical workflow as a structured DAG and introducing a dual-representation state model, SEA reframes the agent's planning challenge into a more robust node classification task. This structured approach deliberately separates high-latency data subsetting from low-latency insight generation, enabling efficient state reuse across conversational turns.

Our evaluation on the newly created GloboMart benchmark demonstrates the efficacy of this approach. The SEA planner achieved 95.65% accuracy in selecting the correct workflow entry-point, resulting in high end-to-end task success and a dramatic reduction in conversational latency for follow-up queries. These results highlight a broader principle for agent design: structuring the environment with domain-specific priors is a powerful mechanism for improving agent performance and reliability, shifting the burden from pure LLM reasoning to strategic optimization within a well-defined process.

While effective, our work has limitations that suggest avenues for future research. The current workflow DAG is static; future systems could learn or dynamically construct these graphs to handle more varied analytical tasks. Our session-based memory model prevents context drift but limits cross-topic reasoning, pointing towards a need for more advanced memory architectures. Finally, with schema retrieval identified as the primary performance bottleneck, targeted improvements in this area represent a clear path toward even higher system accuracy. We believe that further research into co-designing agents and their environments will be critical to unlocking the full potential of LLM-based systems.

References

Amazon Web Services. Amazon s3 (simple storage service). `https://aws.amazon.com/s3/`, 2025. Service first launched in 2006. Accessed: 2025-09-03.

Q. An, C. Ying, Y. Zhu, Y. Xu, M. Zhang, and J. Wang. Ledd: large language model-empowered data discovery in data lakes. *arXiv preprint arXiv:2502.15182*, 2025.

M. Armony, A. Meroño-Peñuela, and G. Canal. How far are llms from symbolic planners? an nlp-based perspective. *arXiv preprint arXiv:2508.01300*, 2025.

H. Chase. LangChain, Oct. 2022. URL `https://github.com/langchain-ai/langchain`.

M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chroma. Chroma db. `https://github.com/chroma-core/chroma`, 2023. Accessed: 2025-09-03.

Databricks. Unity catalog. `https://www.databricks.com/product/unity-catalog`, 2025. Accessed: 2025-09-03.

V. Dibia. Lida: A tool for automatic generation of grammar-agnostic visualizations and infographics using large language models. *arXiv preprint arXiv:2303.02927*, 2023.

R. Ding, S. Han, and D. Zhang. Insightpilot: An llm-empowered automated data exploration system. In *EMNLP 2023. ACL special interest group on linguistic data (SIGDAT)*, 2023.

Encode. Uvicorn. `https://github.com/encode/uvicorn`, 2018. Accessed: 2025-09-03.

Google. Gemini Language Model. `https://deepmind.google/technologies/gemini/`, 2024. Accessed: 2025-04-16.

H. Harkous, I. Groves, and A. Saffari. Have your text and use it too! end-to-end neural data-to-text generation with semantic fidelity. *arXiv preprint arXiv:2004.06577*, 2020.

LangChain Team. Langgraph. `https://github.com/langchain-ai/langgraph`, 2024. Accessed: 2025-09-03.

F. Lei, J. Chen, Y. Ye, R. Cao, D. Shin, H. Su, Z. Suo, H. Gao, W. Hu, P. Yin, et al. Spider 2.0: Evaluating language models on real-world enterprise text-to-sql workflows. *arXiv preprint arXiv:2411.07763*, 2024.

J. Li, B. Hui, G. Qu, J. Yang, B. Li, B. Li, B. Wang, B. Qin, R. Geng, N. Huo, et al. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36, 2024.

X. Liu, S. Shen, B. Li, P. Ma, R. Jiang, Y. Zhang, J. Fan, G. Li, N. Tang, and Y. Luo. A survey of text-to-sql in the era of llms: Where are we, and where are we going? *IEEE Transactions on Knowledge and Data Engineering*, 2025.

C. Packer, V. Fang, S. Patil, K. Lin, S. Wooders, and J. Gonzalez. Memgpt: Towards llms as operating systems. 2023.

S. Ramírez. Fastapi. `https://github.com/tiangolo/fastapi`, 2018. Accessed: 2025-09-03.

N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551, 2023.

N. Shinn, F. Cassano, B. Labash, A. Gopinath, K. Narasimhan, and S. Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. *URL https://arxiv. org/abs/2303.11366*, 1, 2023.

B. Wang, R. Shin, X. Liu, O. Polozov, and M. Richardson. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. *arXiv preprint arXiv:1911.04942*, 2019.

L. Wang, W. Xu, Y. Lan, Z. Hu, Y. Lan, R. K.-W. Lee, and E.-P. Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *arXiv preprint arXiv:2305.04091*, 2023.

P. Wang, Y. Yu, K. Chen, X. Zhan, and H. Wang. Large language model-based data science agent: A survey. *arXiv preprint arXiv:2508.02744*, 2025.

L. Weng, Y. Tang, Y. Feng, Z. Chang, P. Chen, R. Chen, H. Feng, C. Hou, D. Huang, Y. Li, et al. Data-lab: A unifed platform for llm-powered business intelligence. *arXiv preprint arXiv:2412.02205*, 2024.

L. Weng, X. Wang, J. Lu, Y. Feng, Y. Liu, H. Feng, D. Huang, and W. Chen. Insightlens: Augmenting llm-powered data analysis with interactive insight management and navigation. *IEEE Transactions on Visualization and Computer Graphics*, 2025.

Y. Wu, T. Yue, S. Zhang, C. Wang, and Q. Wu. Stateflow: Enhancing llm task-solving through state-driven workflows. *arXiv preprint arXiv:2403.11322*, 2024a.

Z. Wu, Z. Li, J. Zhang, M. Li, Y. Zhao, R. Fang, Z. He, X. Li, Z. Li, and S. Song. Rb-sql: A retrieval-based llm framework for text-to-sql. *arXiv preprint arXiv:2407.08273*, 2024b.

H. Yang, S. Yue, and Y. He. Auto-gpt for online decision making: Benchmarks and additional opinions. *arXiv preprint arXiv:2306.02224*, 2023.

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

X. Zhang, D. Wang, L. Dou, Q. Zhu, and W. Che. Murre: Multi-hop table retrieval with removal for open-domain text-to-sql. *arXiv preprint arXiv:2402.10666*, 2024.

Appendices

A System Implementation Setup Details

We instantiated the SEA methodology into a concrete system for empirical validation. The core components were implemented using the following configuration:

LLM Configuration We employed a two-tiered model strategy to balance reasoning capability with latency. For the strategic, high-level reasoning required by the *Planner Agent* ($\mathcal{P}$), we use the Gemini Pro model Google [2024]. For the more constrained, latency-sensitive tasks performed by the individual tools—including the reranking in `SchemaRetriever` and the code generation in `QuerySynthesizer` and `AIAnalytics`—we utilize the lightweight Gemini Flash model.

Infrastructure The action space $\mathcal{A}$ is realized as a suite of microservices. Each tool is exposed as a distinct API endpoint, developed with FastAPI and served by Uvicorn Ramírez [2018], Encode [2018]. The semantic indexing for the `SchemaRetriever` is implemented using ChromaDB as the vector store Chroma [2023], populated with embeddings generated by a Sentence Transformers model Reimers and Gurevych [2019]. For our research prototype, the Main State (S) and Twin Summary (S') DAGs are managed in-memory as Python dictionaries.

Data Backend The GloboMart dataset is stored as Parquet files on AWS S3 Amazon Web Services [2025], and is accessed for real-time querying via a Databricks environment configured with Unity Catalog Databricks [2025]. The final user-facing conversational interface is also exposed as an API endpoint. The complete, containerized codebase is provided in the supplementary materials to ensure full reproducibility.

B The GloboMart Analytics Dataset & Conversational Query Set

This appendix provides a comprehensive guide to the GloboMart benchmark environment, which was created to rigorously evaluate conversational data analysis systems. The environment consists of a large-scale, realistic e-commerce data warehouse and a corresponding set of 70 multi-turn conversational queries. The dataset has been provided in the supplementary material `https://osf.io/buxma/files/osfstorage?view_only=f7de66430a7b42e0acbc6330ecedd255`.

B.1 Part 1: The GloboMart Data Warehouse

Business Context & Narrative GloboMart is a fictional global retailer with five years of simulated operations. The dataset captures the entire customer journey, enabling analysis across marketing, web engagement, sales, and fulfillment. It is designed to answer business questions such as campaign effectiveness (`Dim_Marketing`, `Fact_Web_Analytics`), user behavior on the website (`Fact_Web_Analytics`), drivers of profitability (`Fact_Sales`), and supply chain efficiency (`Dim_Shipments`).

Generation Methodology The dataset was programmatically generated using a Python-based simulation engine to ensure temporal consistency and analytical depth. The simulation logic, which ran day-by-day over a five-year period, injected realistic business patterns:

- **Seasonality:** Sales volumes for relevant product categories (e.g., 'Electronics') were increased in corresponding quarters (e.g., Q4).
- **Event-Driven Spikes:** Sales were multiplied on weekends and holidays.
- **Customer Popularity:** A power-law distribution ensures a realistic concentration of sales among a small number of customers and products.
- **Correlated Events:** Website conversion events are directly linked to transactional records in `Fact_Sales`.

Schema Architecture: The Star Schema The data warehouse employs a classic star schema, optimized for high-performance analytical queries. This join-heavy architecture features central Fact tables surrounded by descriptive Dimension tables.

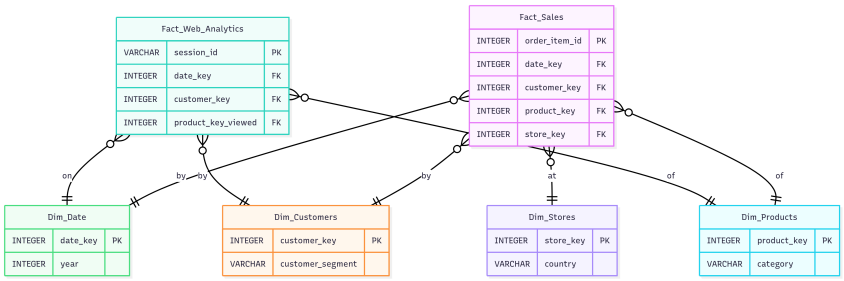

Figure 4: The star schema of the GloboMart data warehouse. Central fact tables (`Fact_Sales`, `Fact_Web_Analytics`) are linked to surrounding main dimension tables that provide descriptive context. Remaining two are similarily linked.

B.2 Part 2: The Conversational Query Set

The benchmark contains 70 queries structured into conversational sessions to test a system's ability to manage context, handle ambiguity, and reuse results.

Join Complexity The query set intentionally tests performance on joins of varying complexity.

- **Single-Table Queries:** e.g., "What is the average session duration?"
- **Two-Table Joins:** e.g., "What were the total sales by product category?"
- **Three/Four-Table Joins:** e.g., "What was the total profit from 'High-Value' customers in 'India' last quarter?"

Multi-Turn Conversational Flows Queries are grouped to mimic natural data exploration.

- **Drill-Downs:** A user starts broad and gets more specific. (e.g., "Total sales last year" → "...broken down by country" → "...just for 'Electronics'.")
- **Topic Switches:** A user finishes one line of inquiry and starts a new one, testing the session lifecycle. (e.g., "Thanks for the sales data. Now show me marketing campaign performance.")

Diverse Analytical Intent & Output The phrasing of queries is designed to elicit a wide range of analytical outputs and test presentation capabilities.

- **Analysis Types:** Queries cover trend analysis, comparative analysis, and hypothetical scenarios.
- **Output Formats:** Phrasing prompts for various outputs including KPI cards, data tables, natural language summaries, and diverse chart types (bar, line, geographic maps).
- **Ambiguity and Corrections:** The benchmark includes natural language phenomena like ambiguous requests ("show me top products") and user corrections ("sorry, I meant...").

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction claim that structuring the agent's environment simplifies the planning task and improves efficiency for conversational data analytics. Our experimental results in Section 6 directly validate this, showing high planner accuracy (95.65%) on its simplified task and a 36% average latency reduction from state reuse.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in the Conclusion (Section 7). We identify the schema retrieval component as the primary performance bottleneck and discuss the scope of generalizability beyond our synthetic, e-commerce-focused benchmark.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper introduces an empirical system and a benchmark; it does not present theoretical results, theorems, or formal proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide comprehensive details of our experimental setup, including the dataset architecture, benchmark design, and evaluation metrics in Section 4 and Appendix B. Further implementation details, including the specific LLM used and system architecture, are provided in Appendix A.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide open access to the GloboMart dataset, the conversational benchmark, and the source code for the SEA framework. A URL is provided in a footnote in Section 4, and detailed documentation is in the supplementary material and Appendix B.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Full details of the experimental setting, including the LLM used for the Planner, system architecture, and evaluation protocols, are provided in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: Due to the high cost of running the full 100-query benchmark and conducting human evaluations, we report results from a single comprehensive run. We acknowledge that reporting error bars over multiple runs would strengthen our results, but this was computationally prohibitive for the current work.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide details on the computational resources used for our experiments, including the type of GPU, the LLM API used for the Planner, and average query execution times, in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and believe our research, which focuses on a synthetic dataset and system development for data analytics, conforms to its principles.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader societal impacts in the Conclusion. Positive impacts include democratizing data access for non-technical users, while potential negative impacts include the risk of users over-relying on or misinterpreting AI-generated insights.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our work does not release a new pre-trained language model or a scraped dataset with high-risk content. The released assets are a synthetic dataset generated without real user data and source code for our system, which we deem to pose no direct high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We credit the open-source libraries and specify the large language model used in our work in Appendix A. The terms of use for the proprietary LLM API were respected during our research.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We provide comprehensive documentation for our new assets, the GloboMart dataset and conversational benchmark, in Appendix B. The documentation includes the generation methodology, schema, and query design principles, and is released alongside the assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [Yes]

Justification: Details of our human evaluation protocol, including the guidelines and criteria given to the expert evaluators for assessing the end-to-end quality of system responses, are provided in Appendix A. No external crowdsourcing was used.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our research involved expert human evaluators (the authors) assessing the quality of system-generated text and visualizations. This task poses minimal risk to participants and, in our institutional context, did not require formal IRB approval.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The core of our proposed SEA framework is an LLM-based planner, as described throughout the paper (e.g., Section **??**). The LLM's role as the agent's reasoning engine is a central and non-standard component of our system's architecture.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.