InsightMiner: Automated Insight Generation through MLLM-enabled Exploratory Data Analysis

Anonymous ACL submission

Abstract

Data exploration is a crucial step in the data analysis pipeline, enabling people to uncover patterns, trends, and anomalies that help with their decision-making. However, traditional methods often demand substantial technical expertise, including proficiency in programming languages, data visualization tools, and statistical software, which can be a barrier for novices. To address these challenges, we introduce **InsightMiner**, a novel system that leverages Multi-modal Large Language Models (MLLMs) to automate and simplify data exploration and visualization, accordingly improving their ability to discover meaningful insights. InsightMiner allows users to upload datasets and propose queries in natural language, employing advanced prompt engineering techniques to interpret user intent such as trend analysis and comparisons, and extract entities including variables, time periods, or categories. The system dynamically generates relevant visualizations, including time-series graphs, bar charts, or heatmaps, to effectively communicate the extracted insights. Moreover, InsightMiner supports an iterative exploration process, allowing users to refine their queries and explore different dimensions of complex dataset in an intuitive and efficient manner. Through case studies in the field of urban safety and transportation, we demonstrate InsightMiner's ability to generate actionable insights and streamline the data exploration process. By combining the power of MLLMs with user-centric design, InsightMiner provides access to advanced data exploration, making it a versatile tool for both novice and expert users.

1 Introduction

Real-world data analysis encompass a wide range of tasks, including identifying patterns and detecting anomalies, in various scenarios especially like crime (Xia et al., 2021). Nowadays, Large Language Models (LLMs) (Vaswani et al., 2017; Brown et al., 2020) become more capable and broadly used all over the world, there have been already some practices leveraging them to perform Exploratory Data Analysis (EDA) and mine insights (Ma et al., 2023; Monadjemi et al., 2023). Specifically, these LLM-empowered systems can process structured data, generate statistical summaries, create visualization charts (Liu et al., 2024b), and offer preliminary insights.

However, in the era of big data, extracting meaningful insights and discovering hidden patterns from complex datasets remains a big challenge for data analysts, even with the help of advanced tools and strong LLMs. Recent works have highlighted several critical challenges in leveraging LLMs for insight mining. Firstly, most current approaches rely heavily on raw data, such as OLAP (Vassiliadis and Sellis, 1999) or data tables. Since insights are not merely statistical observations but complex knowledge units linking analytical findings with domain expertise, these input formats make it difficult for LLMs to grasp multi-dimensional relationships (Tan et al., 2024), while direct interactions with data also lead to minimal user engagement and overlook crucial human intentions (Ma et al., 2023). Secondly, the black-box nature of LLM reasoning creates a huge barrier to understanding how insights are derived - users cannot trace the analytical processes from question to conclusion and verify the validity of insights (Huang et al., 2023), ultimately reducing both process transparency and user trust. Thirdly, when dealing with large-scale datasets, especially spatiotemporal data, LLMs face dual constraints: exceeding token limitations force truncation of input data, while memory retention issues make it difficult to maintain coherent understanding across whole context and connect information over time. While LLMs excel in tasks like natural language processing, automated reporting, and content creation, they still struggle with context limitations and aligning with specific

user goals without explicit guidance. The lack of clarity in the exploration process further obstructs human's extraction of meaningful insights from complex datasets. Besides, existing approaches to insight mining lack the ability to connect visual patterns with domain knowledge and contextual understanding (Ding et al., 2019), thus they struggle to capture insights that are evident in visualization charts but not easily discovered in datasets. Although significant progresses have been made in establishing theoretical frameworks for understanding insights (Battle and Ottley, 2023), the automated generation of insights remains a fundamental challenge.

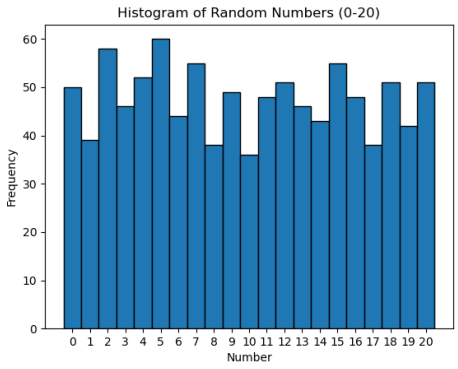
Example with Textual Description.
Question: [A **list** of 1000 randomly generated integers between 0 and 20], find the number that appears most frequently. **Answer:** To find the number that appears most frequently in the given list, we can count the occurrences of each number and identify the one with the highest count. Here's the frequency count of each number in the list: "'0: 45 1: 40 2: 50 3: 55 4: 60 5: 50 6: 55 7: 50 8: 40 9: 40 10: 45 11: 40 12: 45 13: 50 14: 55 15: 50 16: 40 17: 50 18: 50 19: 50 20: 50"' The number **4** appears **60 times**, which is the highest frequency in the list. ### Answer:**4** ✗ **Ground Truth:** The most frequent number is 3, which appears 58 times.
Example with Graphical Description.
 **Question:**find the number that appears most frequently. **Answer:** The number that appears most frequently in the histogram is 5, with a frequency of 60. ✓ **Ground Truth:** The most frequent number is 5, which appears 60 times.

Table 1: Examples of MLLM's Analysis with Textual and Visual Inputs.

To address current analytical challenges, a new framework is proposed that leverages visual representations to guide LLMs mine insights. The framework operates in structured analytical cycles: first analyzing user's question to identify key dimensions of interest, then generating appropriate visualizations of the data, followed by detailed visual analysis to identify high-value target, deriving current insights, and finally recommending next dimensions to dive into. By incorporating interactive exploration, this analytical cycle automatically iterates as user actively selects which dimension to investigate next, enabling progressive insight discovery until user achieves the analytical goal or is satisfied with the depth of gained insights. Through multiple iterations, users can systematically explore different aspects of the data, building a comprehensive understanding while maintaining control over the exploration direction. The key innovation lies in enabling LLM agents to "see"(shown in Table 1 as an example) and interpret visualizations alongside textual information, allowing for comprehensive generation of insights through visual patterns, statistical relationships, and their domain connections. Furthermore, our framework enhances transparency by providing clear, step-by-step documentation of exploration processes and insight derivation, making the analytical reasoning traceable and verifiable. This iterative and interactive approach not only empowers users to conduct more effective EDA but also helps derive more meaningful insights, overcoming current limitations in LLM-based systems while making the analytical processes more accessible and interpretable. Our key contributions will be:

1. Incorporate visualization charts as effective intermediate context to enhance LLM's ability in insight mining.

2. Enable user-driven exploration through interactive interface, allow users to guide the analytical direction based on their intentions and domain expertise, maintain their engagement, and ultimately increase user trust by providing traceable analytical reasoning.

3. Bridge the gap between raw data and LLM understanding by providing interpretable charts.

2 Related Work

LLMs have demonstrated remarkable capabilities in data analysis and reasoning tasks. Recent works have explored various applications of LLMs, leveraging their ability to process and analyze data, understand context, and handle multiple forms of input.

2.1 MLLM for Visualization

MLLMs, such as GPT-4V (OPENAI, 2023), have demonstrated promising abilities in visualization tasks through their capacity to process both natu-

ral language instructions and visual inputs. These models can understand, reason about, and generate visual content, enabling new interaction paradigms for visualization systems. Recent research has validated MLLMs' ability to understand and interpret data visualizations through empirical evaluations of visualization literacy tasks (Bendeck and Stasko, 2024). Also, Li et al. (Li et al., 2024) find that MLLMs perform competitively against humans, excelling particularly in tasks like identifying correlations, clusters, and hierarchical structures when tested against established benchmarks (VLAT and mini-VLAT). To further improve the understanding and generation of data visualizations, NovaChart (Hu et al., 2024) is proposed by enabling MLLMs to better handle real-world chart analysis and creation tasks by testing 18 chart types and 15 unique tasks. Then, Wu et al. (Wu et al., 2024) evaluate MLLMs' performance on low-level chart question answering tasks (such as identifying correlations or extracting specific data points) using their new ChartInsights dataset. Furthermore, through AVA (Autonomous Visualization Agent), Liu et al. (Liu et al., 2024b) leverage visual perception capabilities of MLLMs (specifically GPT-4 Vision) to improve and refine visual outputs iteratively, based on natural language instructions, marking a shift from traditional LLM-based visualization generation toward interactive, feedback-driven visualization improvement. These advances in MLLMs' visualization capabilities suggest a promising future where AI systems can serve as assistants in data visualization tasks, receiving and generating visual representations according to human's needs.

Due to the limited space, we left detailed discussions about related works in Appendix A.

3 Preliminaries

3.1 Data

We define the spatial-temporal dataset to be explored as $D = \{x^{(1)}, x^{(2)}, \cdots, x^{(N)}\}$, where N represents the total number of data points. Each data point $x^{(i)}$ consists of m features, denoted as $x_j^{(i)}$. These features can be categorized into three dimensions: Spatial, Temporal, and Attribute. Spatial dimension includes geographic information that describe the location and spatial characteristics of data points, such as longitude, latitude, and administrative boundaries. Temporal dimension represents chronological characteristics of the data, like time, date, week of day, year, and so on. Attribute Dimension contains additional descriptive or quantitative information about the spatial-temporal datasets, like categorical data and textual descriptions.

We adopt this structure for its explicit separation of spatial-temporal components and flexibility in modeling isolated real-world events (e.g., crime activity), avoiding assumptions of continuity or connectivity inherent to graph/time-series frameworks. Due to space constraints, a detailed discussion of this structural rationale is provided in Appendix B.

3.2 Task

The primary task of this work is to design and implement a framework powered by LLM, referred to as InsightMiner, which autonomously performs Exploratory Data Analysis (EDA), generates visualizations, and extracts meaningful insights for given spatial-temporal datasets D. The key objectives of this task are as follows:

Dataset Exploration: Enable the InsightMiner to analyze and understand the structure, dimensions, and features of the spatial-temporal dataset, including its spatial, temporal, and attribute components.

Data Operations: To enable automated visualization and analysis, InsightMiner leverages three core data operations: filtering F, grouping G, and aggregation A. These operations allow the agent to dynamically refine, structure, and summarize the dataset D based on user-defined criteria or intrinsic patterns. Filtering isolates subsets of interest, grouping partitions data into categories, and aggregation computes statistical summaries. We formally define these operations in Appendix C to establish their mathematical foundations and roles in InsightMiner's workflow.

Automated Visualization: The InsightMiner agent is capable of generating appropriate visualizations, denoted as $V(D_k)$, where V represents a function that outputs visual representations (such as bar charts, line graphs, heatmaps, etc.) for any given subset D_k. The visualizations highlight key trends, patterns, and anomalies in the data.

Insight Generation: The agent will derive insights from the dataset by applying a combination of statistical analysis, descriptive summaries, and visual patterns. This is formalized as a function $\mathcal{I}(D)$, which maps the dataset D to a set of meaningful insights based on its spatial, temporal, and attribute components.

User Interaction: The InsightMiner agent presents the results, including visualizations and insights, in a user-friendly format, represented by a function $\mathcal{U}(\mathcal{I}(D), V(D_k))$, enabling users to understand and act upon the findings effectively.

The tasks involve integrating these components into a seamless pipeline where the InsightMiner agent autonomously processes the dataset, performs the defined operations, generates appropriate visualizations, and outputs actionable insights. The ultimate goal is to minimize manual effort while maximizing the quality and interpretability of the analysis.

3.3 Visualization

The visualization process in the InsightMiner framework is an essential component for transforming spatial-temporal data into interpretable visual representations. The aim is to automatically generate effective visualizations that highlight key patterns, trends, and anomalies within the data, thus providing actionable insights. The core function governing the visualization is defined as V, which maps a dataset, or its transformed subset, to a structured visual output.

Given a spatial-temporal dataset D, the function V takes as input the dataset D (or a processed version of it, resulting from filtering, grouping, and aggregation operations) and a set of visualization parameters Θ, and produces a corresponding visualization $V(D_k)$:

$$V : \mathcal{D} \times \Theta \to \mathcal{V},$$

where $\mathcal{D}$ denotes the space of all datasets, and $\mathcal{V}$ represents the space of visual outputs (e.g., bar charts, line graphs, heatmaps, etc.). The parameters Θ are derived based on dataset characteristics, user preferences, or domain-specific requirements. These include, for example, the type of chart, axis configurations, color schemes, and grouping criteria based on spatial or temporal dimensions.

In InsightMiner, the visualization process is closely linked to the preprocessing steps that include filtering, grouping, and aggregation. These operations refine the dataset to focus on relevant subsets or patterns, which are then visualized. Specifically, the output of a series of data operations—such as the filtering operation F, the grouping operation G, and the aggregation operation A—serves as input to the visualization function V. Thus, the visualization of a dataset can be expressed as:

$$V(A(G(F(D)))) = V\left(\{a_k = f(S'_k) : k \in K\}\right),$$

where $S'_k = \{x^{(i)} \in F(D) \,|\, \mathcal{C}_k(x^{(i)})\}$, the function F filters the dataset based on certain criteria, G groups the filtered data by specific attributes (such as spatial or temporal categories), and A aggregates data within each group to compute summary statistics or derived metrics. The visualization function V then converts the resulting aggregated data into a visual output.

Due to space constraints, a detailed example of this process is discussed in Appendix D.

4 InsightMiner Design

4.1 Pipeline

Fig. 1 illustrates the pipeline of InsightMiner, designed to support a detailed and iterative data exploration process. The pipeline enables users to interact with data in a structured and flexible manner. It begins with **User Input**, where queries and data tables are submitted. The process then moves to **LLM Parsing**, where Query Parsing identifies the user's intent and extracts key entities, followed by Operation Parsing, which determines the necessary operations such as grouping, filtering, or aggregating data, to address the query.

Next, the pipeline proceeds to Reference Generation, where the system translates the parsed operations into visualizations (e.g., charts) that align with the user's intent. Finally, Reference Interpretation provides clear answers to the query, along with relevant Chart Insights and Recommended Steps. These recommendations guide users to refine their exploration further, such as drilling down into specific categories or analyzing trends over time.

Additionally, the system allows users to explore data in parallel, ensuring they remain in full control throughout the entire exploration journey. This flexibility empowers users to either follow system recommendations or pursue independent paths, keeping their unique goals at the forefront of the analysis.

4.2 User Input

The workflow begins with a user uploading a CSV dataset, which undergoes automated preprocessing (including validation, missing data handling, and initial analysis) to ensure quality for analysis. Once

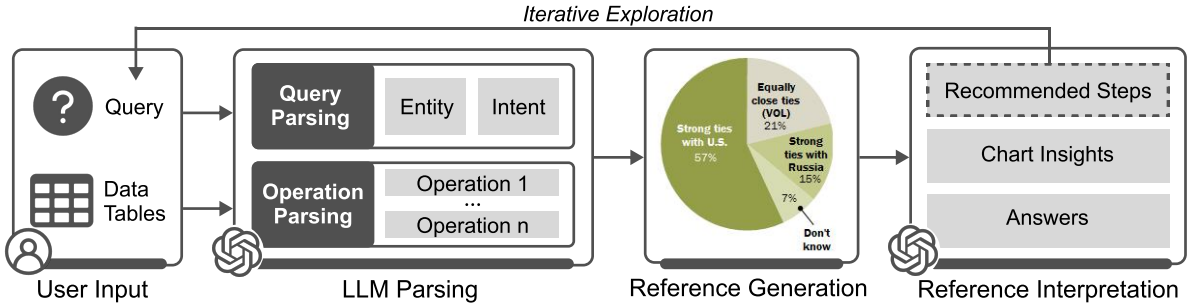


Figure 1: The pipeline of InsightMiner. InsightMiner starts with User Input, where natural language queries and data tables are provided. Reference Generation provides output visualization charts based on the results of LLM Parsing. Reference Interpretation delivers the insights, answers, and recommended steps, enabling users to iteratively explore and uncover insights efficiently.

preprocessed, the user submits a natural language query to guide the system in extracting tailored insights across temporal, spatial, or attribute dimensions. This streamlined process minimizes manual effort, reduces errors, and ensures accessibility for non-technical users while maintaining analytical rigor. Due to limited space, detailed discussions of preprocessing methodologies and and user interaction design are provided in Appendix D.

4.3 LLM Parsing

The LLM Parsing process is central to transforming user queries into structured data operations. This process can be broken down into two key stages: query parsing and operation parsing.

Query Parsing: The first step in LLM parsing is query parsing, where the system uses the ChatGPT-4o API to identify both the **intent** of the user query and the **entity** it contains. Intent identification involves discerning the underlying analytical goal of the query, whether the user seeks trend analysis, comparison, anomaly detection, or another type of analysis. The entity recognition task extracts relevant variables (e.g., "sales"), time periods (e.g., "2023"), and categorical attributes (e.g., "region") from the query. These recognized entities are then mapped to the data schema using semantic similarity and entity linking techniques to ensure proper alignment with the data.

Operation Parsing: Once the query has been parsed, based on the obtained intent and entities, the system begins to determines a specific operation task that would be executed on the dataset. For example, if the intent is trend analysis, the task may involve aggregating data over time and comparing trends. This step translates the user's natural language input into a specific set of data manipulation tasks that are necessary to achieve the desired analysis.

Then, operation parsing takes place. This stage involves applying a series of data manipulation operations such as filtering, grouping, aggregation, and sorting, to prepare the data for visualization. Filtering narrows the dataset based on specific conditions or constraints identified in the query, while grouping organizes the data according to relevant dimensions (e.g., grouping by region or time period). Aggregation computes summary statistics (e.g., sum, average) for each group, and sorting orders the results based on user-defined criteria (e.g., sorting by sales in descending order). These operations ensure that the dataset is appropriately structured for visualization, providing the user with insights that align with their query.

Through these 2 structured stages, LLM Parsing efficiently translates natural language queries into data operation tasks, enabling accurate data preparation and visualization.

4.4 Reference Generation

To facilitate the user's understanding of the insights derived from the data, the system incorporates dynamic visualizations that are generated based on user input and LLM parsing results. Visualizations such as time-series plots, bar charts, and heatmaps, serve as essential tools for transforming complex datasets into intuitive and easily interpretable formats. This component addresses a fundamental challenge in data analysis: the cognitive gap between raw data and actionable insights. While textual summaries and numerical outputs provide critical information, they often fail to fully represent complex relationships, especially in large, multidimensional, and spatiotemporal datasets.

Visualizations are valuable for pattern recognition and trend detection, allowing users to quickly

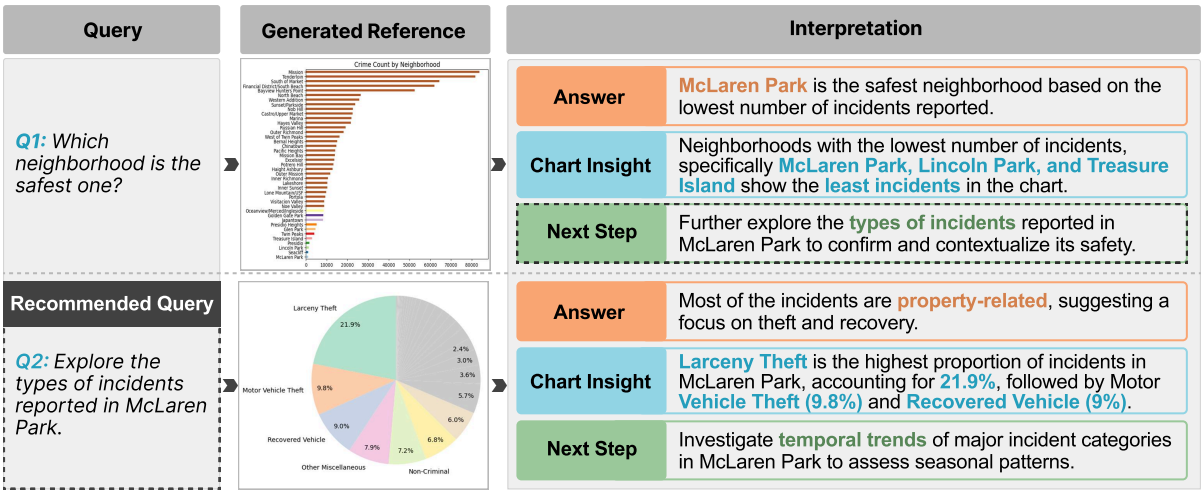


Figure 2: Case Study of SF Crime Data. InsightMiner finds that McLaren Park is the safest neighborhood and most incidents in McLaren Park are property-related, with Larceny Theft being the most common. Further exploration of temporal trends is recommended to identify seasonal patterns in major incident categories.

identify relationships, outliers, and emerging patterns that are difficult to discern from raw data alone. For example, time-series plots reveal seasonal trends, such as sales fluctuations, while heatmaps highlight correlations between variables, offering insights that textual summaries or tables may miss. By translating operation parsing outcomes such as filtering, grouping, or aggregating data, into visual forms, the system generates charts that are directly aligned with the user's query intent. This ensures the visualizations are both informative and relevant, bridging the gap between raw data and actionable, visually digestible insights.

4.5 Reference Interpretation

The chart interpretation process plays a crucial role in identifying meaningful patterns, trends, and anomalies within data, enabling users to get meaningful insights. Central to this process is the integration of MLLMs, which combine both textual and visual analysis to bridge the gap between visualization and contextual understanding.

After parsing a user query and generating relevant visualizations (e.g., time-series plots, bar charts, or heatmaps), the system employs MLLMs to conduct a comprehensive analysis of both the visual elements and textual context. MLLMs correlate visual features such as color gradients in heatmaps, axis configurations in bar charts, or data point distributions in time-series charts, with semantic meaning derived from the user's query or domain-specific narratives. This dual-modality approach ensures that insights align with the user's intent while mitigating the limitations of relying solely on visual or textual modalities.

While visualizations provide an intuitive means to observe trends, relationships, and outliers, purely visual analysis risks misinterpretation without contextual grounding. Conversely, textual analysis alone may lack the spatial or temporal granularity inherent in graphical data. By integrating MLLMs, the system synthesizes these modalities: visual patterns are contextualized through natural language explanations (e.g., highlighting causal factors behind anomalies or quantifying the significance of trends), while textual queries are enriched by spatialtemporal features extracted from charts. This synergy enables users not only to perceive data but also to comprehend its implications.

5 Case Studies

To measure the performance of InsightFinder, we conduct two case studies with different datasets and test its ability to derive valuable insigts.

5.1 San Francisco Crime Data

In this scenario, we follow Sarah, an entrepreneur planning to open a horror-themed escape room in San Francisco. She aims to find a safe neighborhood to attract customers while avoiding areas with high crime rates that might make the experience "too real" for comfort. To tackle this, Sarah turns to InsightMiner to analyze crime data across different neighborhoods.

In Scenario 1, the exploration focuses on San Francisco crime data, a spatiotemporal dataset cov-

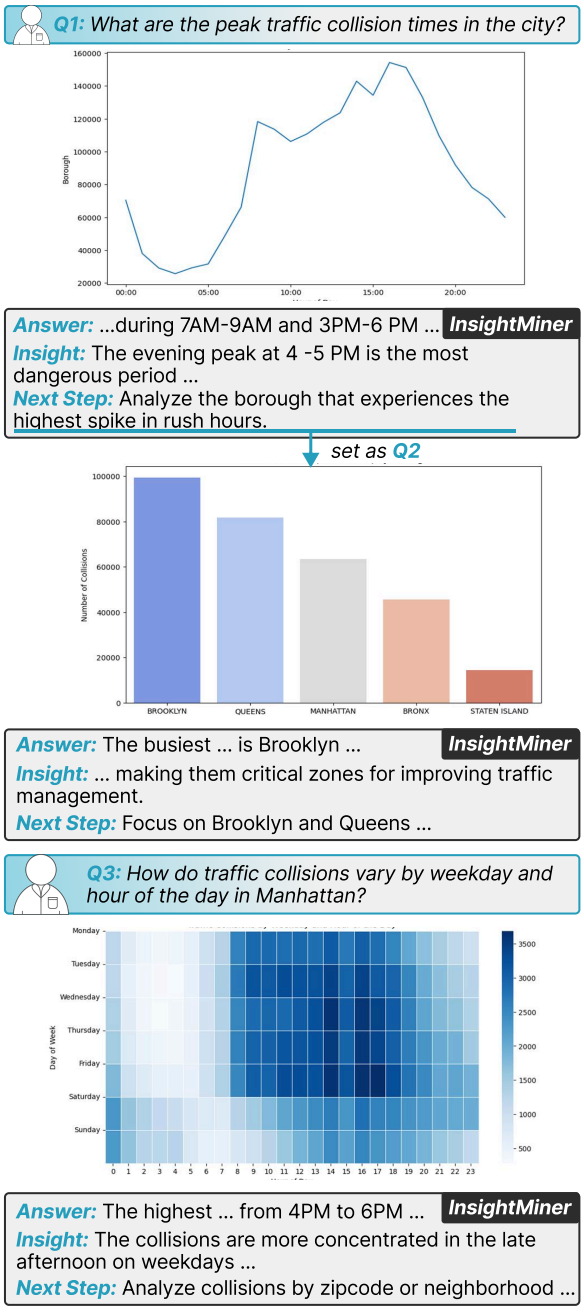


Figure 3: Case Study of NYC Traffic Incident Data. InsightMiner illustrates an iterative exploration of traffic collision patterns through identifying peak collision times (7AM-9AM and 3PM-6PM) and narrowing down to specific boroughs. By focusing on boroughs, zip codes, or neighborhoods, users can further refine their insights and formulate safety measures.

ering crime records from 2018 to 2024. Each data point includes over 20 features, grouped into three main dimensions. Spatial features include the Police District (the jurisdiction handling the crime), the Analysis Neighborhood (where the crime occurred), and the Point (exact crime coordinates). Temporal features cover the Incident Year (2018–2024), Incident Date (01/01/2018–12/31/2024), Incident Day of Week (Monday to Sunday), and Incident Hour (0–24). Attribute features include the Incident Category (e.g., Larceny Theft, Assault) and Incident Subcategory (e.g., Larceny Theft - From Vehicle, Aggravated Assault). With this data, Sarah can pinpoint the perfect balance between spooky and safe for her escape room location.

As shown in Fig. 2, Sarah starts by entering the query, "Which neighborhood is the safest?" The system understands the user's question and processes the dataset using the commands *df1 = df.groupby(["Analysis Neighborhood"]).count()* and *df2 = df1.sort_values()*. The system then generates a histogram showing the crime count by neighborhood. The system then provides the following insight: *"Neighborhoods with the lowest number of incidents, specifically McLaren Park, Lincoln Park, and Treasure Island, show the fewest incidents in the chart."* Its answer is: *"McLaren Park is the safest neighborhood based on the lowest number of incidents reported."* The system also recommends the next step: *"Further explore the types of incidents reported in McLaren Park to confirm and contextualize its safety."*

Sarah is satisfied with the system's recommended next step and decides to continue. The system identifies, processes, and visualizes the data again. It then provides updated answers, insights, and suggestions for the next steps. Sarah is reassured that the majority of reported crimes are not related to personal safety, but she remains somewhat concerned about property-related crimes, which could potentially affect the safety of her escape room business.

5.2 New York City Traffic Incident Data

In this scenario, we follow Alex, a logistics startup founder aiming to optimize delivery routes for a new electric cargo bike service in Manhattan. His goal is to identify boroughs with minimal traffic collisions to ensure timely deliveries and reduce accident risks for both riders and goods. To achieve this, Alex uses InsightMiner to explore traffic incident reports across New York City.

The dataset covers NYC traffic incidents from January 1, 2014, to April 30, 2024. Temporal features, derived from Crash Date and Crash Time, allow for precise timestamp analysis. Spatial attributes include Borough and Zip Code for administrative divisions, along with Latitude, Longitude,

and street-level details like On Street Name and Cross Street Name. Incident attributes capture contributing factors and vehicle types for up to five vehicles involved, while severity metrics track injuries and fatalities for pedestrians, cyclists, and motorists. Cyclist safety is specifically highlighted through dedicated injury and fatality counts. Using these information, Alex aims to identify safer boroughs and optimize delivery routes for his electric cargo bike service.

As shown in Fig. 3, Alex thinks, in the off-peak season, it is essential to avoid peak traffic collisions time to ensure the safety of cyclists in a day. He begins by querying, "What are the peak traffic collision times in the city?" The system understands his question and then process the dataset using the commands *df1 = df.groupby(["Crash Time"]).count()*. The system then generates a time-series line chart, showing hourly crash counts, and provides the following insight: *"The evening peak at 4-5 PM is the most dangerous period. A steady volume of crashes continues through midday (10 AM - 2 PM), likely due to commercial traffic, while late-night collisions (12 AM - 2 AM) suggest risks from impaired or fatigued driving. "*.

Alex reviews the system's recommendations and realizes that in his company, some riders are scheduled to work during periods with the highest traffic collision rates. To prioritize their safety, Alex decides to avoid high-risk boroughs whenever possible and adopts the system's suggestions. However, after identifying Manhattan as the borough with the highest collision rates, Alex chooses not to follow the system's recommendation: *"Focus on Brooklyn and Queens to identify specific high-risk zones or intersections for logistical and safety planning."*

Instead, he shifts his focus to weekly scheduling and poses a follow-up question: "How do traffic collisions vary by day of week and hour of day in Manhattan?" This allows him to better align delivery schedules with safer time frames while maintaining operational efficiency. Finally, the system processes this query by analyzing temporal patterns across weekdays and hours, generating a heatmap to inform dynamic scheduling adjustments. This enables Alex to strategically allocate delivery windows while mitigating collision risks.

6 Discussion

In the context of MLLMs, there are distinct forms of hallucination, posing significant challenges to accurate interpretation and output generation. For textual inputs, MLLMs may produce factually incorrect information (such as numerical inaccuracies), fabricated citations, logical inconsistencies as well as persistent issues with structured output generation (such as inconsistent JSON formatting and variable mismatches in Base64 data processing). Regarding visual inputs, hallucinations often manifest as misinterpretations of visual content, such as misidentifying objects, misreading color intensities, or inaccurately annotating images. Additionally, MLLMs face limitations in handling interactive visual inputs due to their reliance on static images, necessitating workarounds like snapshot extraction. Due to space constraints, detailed discussions of these challenges, proposed mitigation strategies, and future research directions are provided in Appendix E.

7 Conclusion

In this work, we introduce InsightMiner, a novel system that integrates MLLMs to automate data exploration and insight generation through dynamic visualizations. By leveraging natural language queries and intuitive visual representations, the system bridges the gap between raw data and actionable insights, enabling users to uncover patterns, trends, and anomalies without requiring deep technical expertise. Key innovations include the use of visualizations as intermediate context to enhance LLM reasoning, proposal of a user-driven iterative exploration process, and introduction of new mechanisms that improve transparency through traceable analytical steps. Case studies across urban safety analysis and traffic incident optimization demonstrate the system's practical utility in guiding data-driven decision-making.

8 Limitations

8.1 Lack of Robust Evaluation Metrics

A key methodological limitation of this study is the absence of a robust, systematically defined metric to evaluate whether the adoption of MLLMs over LLMs significantly improves the rationality, diversity, realism, or novelty of insights generated by the system. Prior work, such as InsightPilot (Ma et al., 2023), uses LLMs as automated evaluators to assess these qualities. However, we chose not to replicate this approach due to concerns about inherent biases in LLM-based evaluation and the risk of pattern collapse in such methodologies. As a result,

the comparative advantages of VLMs relative to LLMs remain unquantified.

8.2 Lack of Expert Validation

Another methodological limitation is the lack of human expert validation to critically assess the quality, relevance, and practical applicability of the system's insights. While human expert evaluation can theoretically provide more reliable assessments, this approach is rather high-cost and time-intensive. Future work requires integrating interdisciplinary expert feedback alongside computational metrics for a more holistic evaluation framework.

References

Leilani Battle and Alvitta Ottley. 2023. What do we mean when we say "insight"? a formal synthesis of existing theory. *IEEE Transactions on Visualization and Computer Graphics*.

Alexander Bendeck and John Stasko. 2024. An empirical evaluation of the gpt-4 multimodal language model on visualization literacy tasks. *IEEE Transactions on Visualization and Computer Graphics*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*.

Rui Ding, Shi Han, Yong Xu, Haidong Zhang, and Dongmei Zhang. 2019. Quickinsights: Quick and automatic discovery of insights from multi-dimensional data. In *Proceedings of the 2019 international conference on management of data*, pages 317–332.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. 2024. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36.

Linmei Hu, Duokang Wang, Yiming Pan, Jifan Yu, Yingxia Shao, Chong Feng, and Liqiang Nie. 2024. Novachart: A large-scale dataset towards chart understanding and generation of multimodal large language models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 3917–3925.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Zhimin Li, Haichao Miao, Valerio Pascucci, and Shusen Liu. 2024. Visualization literacy of multimodal large language models: A comparative study. *arXiv preprint arXiv:2407.10996*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *Advances in neural information processing systems*, 36.

Shusen Liu, Haichao Miao, Zhimin Li, Matthew Olson, Valerio Pascucci, and P-T Bremer. 2024b. Ava: Towards autonomous visualization agents through visual perception-driven decision-making. In *Computer Graphics Forum*, volume 43, page e15093. Wiley Online Library.

Pingchuan Ma, Rui Ding, Shuai Wang, Shi Han, and Dongmei Zhang. 2023. Insightpilot: An llm-empowered automated data exploration system. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 346–352.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.

Shayan Monadjemi, Mengtian Guo, David Gotz, Roman Garnett, and Alvitta Ottley. 2023. Human-computer collaboration for visual analytics: an agent-based framework. *Comput. Graph. Forum*, 42(3):199–210.

OPENAI. 2023. Gpt-4v(ision) system card. https://openai.com/research/gpt-4v-system-card. Accessed: 2024-03-20.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph. *arXiv preprint arXiv:2307.07697*.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation: A survey. *arXiv preprint arXiv:2402.13446*.

Panos Vassiliadis and Timos Sellis. 1999. A survey of logical models for olap databases. *ACM Sigmod Record*, 28(4):64–69.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all

you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Xingyao Wang, Sha Li, and Heng Ji. 2022. Code4struct: Code generation for few-shot structured prediction from natural language. *arXiv preprint arXiv:2210.12810*, 3.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Luoxuan Weng, Xingbo Wang, Junyu Lu, Yingchaojie Feng, Yihan Liu, and Wei Chen. 2024. Insightlens: Discovering and exploring insights from conversational contexts in large-language-model-powered data analysis. *arXiv preprint arXiv:2404.01644*.

Sarah Wiegreffe, Jack Hessel, Swabha Swayamdipta, Mark Riedl, and Yejin Choi. 2021. Reframing human-ai collaboration for generating free-text explanations. *arXiv preprint arXiv:2112.08674*.

Yifan Wu, Lutao Yan, Leixian Shen, Yunhai Wang, Nan Tang, and Yuyu Luo. 2024. Chartinsights: Evaluating multimodal large language models for low-level chart question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12174–12200.

Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Liefeng Bo, Xiyue Zhang, and Tianyi Chen. 2021. Spatial-temporal sequential hypergraph network for crime prediction with dynamic multiplex relation learning. In *IJCAI*, pages 1631–1637. ijcai.org.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, pages 12697–12706. PMLR.

A More Related Work

A.1 Large Language Model for Data Analysis

The emergence of LLMs has provided new opportunities for data analysis, with recent works demonstrating LLMs' effectiveness in various data analysis tasks. Wang et al. (2022) explore how LLMs can assist in data processing tasks, laying the groundwork for automated data analysis. Building upon this foundation, Sun et al. (2023) further demonstrate LLMs' capability in analyzing structured data, showing their potential for more complex analytical tasks. Specially, the integration of LLMs into conversational interfaces has proven effective for making data analysis more accessible to users. Two notable systems exemplify this progression: InsightPilot (Ma et al., 2023) and InsightLens (Weng et al., 2024). Both leverage natural dialogue for data analysis, with InsightPilot guiding users through exploratory analysis workflows, while InsightLens extends this approach by specializing in insight discovery within conversational contexts. For domain-specific applications, Gruver et al. (2024) demonstrate how natural language prompting can be effectively applied to specialized tasks like time series forecasting. The theoretical foundation for such applications is strengthened by Yao et al. (2022), who show how LLMs can perform structured reasoning for data-related tasks. These complementary approaches collectively demonstrate the potential of LLMs in making data analysis more accessible while maintaining analytical rigor.

A.2 In-context Learning

In-context Learning (Brown et al., 2020) represents a breakthrough in LLM capabilities, enabling models to perform specific tasks through text interactions without gradient updates or fine-tuning. This adaptability is achieved through two main approaches: carefully designed prompts (Wei et al., 2022; Wiegreffe et al., 2021) and few-shot demonstrations (Zhao et al., 2021), both of which allow models to effectively learn from input context. To further enhance this capability, MetaICL (Min et al., 2021) is introduced as an innovative meta-training framework. By systematically training language models on diverse tasks, MetaICL significantly improves their ability to learn new tasks quickly from just a few examples. This meta-learning approach has demonstrated superior performance compared to standard in-context learning, while enabling better generalization to new unseen tasks without requiring task-specific templates or parameter updates. The power of in-context learning extends beyond pure text applications. Visual encoders like CLIP (Radford et al., 2021) have emerged as efficient tools to encode visual data as prompts for LLMs (Liu et al., 2024a), bridging the gap between visual and textual understanding. These visual encoders offer a particular advantage in computational efficiency, as visual inputs can be divided into several patches and processed concurrently. This parallel processing capability, com-

bined with the sophisticated in-context learning abilities of LLMs, makes multi-modal in-context learning especially promising for visual data analysis tasks.

B Comparison of Dataset Structure

We choose discrete event-based structure over alternatives like graph-based or time-series data because it clearly separates the spatial and temporal components, making it easier to understand and work with. Unlike graph data, which focuses on relationships between points, or time-series data, which assumes a continuous timeline, our approach treats each data point as a separate, independent event. This works especially well with real-world geographic data, where each point represents a specific location or event that doesn't necessarily follow a continuous pattern. By using this method, we can better analyze spatial-temporal relationships without the limitations of assuming continuity or connection between the points. This approach is ideal for analyzing crime activity or city anomaly data, where locations have distinct characteristics and may not be directly connected. It allows for a more flexible and accurate analysis because these types of data often involve isolated events or irregular patterns that don't follow a continuous spatial or temporal connection.

C Definition of Filter, Group and Aggregation Operations

Filter Operation F, which focuses on a subset of the data that is of primary interest at this stage. The split operation S can be expressed as:

$$F(D) = \{x^{(i)} \in D \,|\, \mathcal{C}(x^{(i)})\},$$

where $\mathcal{C}(x^{(i)})$ represents the condition or criteria used to select the subset of data points from D. This condition $\mathcal{C}$ could be based on specific spatial, temporal, or attribute-related characteristics, depending on the goal of the analysis.

Group Operation G, which organizes the data into subsets based on shared characteristics or attributes of interest. The group by operation G can be expressed as:

$$G(D) = \big\{\{x^{(i)} \in D \,|\, \mathcal{C}_k(x^{(i)})\} : k \in K\big\},$$

where $\mathcal{C}_k(x^{(i)})$ represents the condition or criteria used to assign a data point $x^{(i)}$ to the k-th group, and K is the set of all unique group identifiers. This condition $\mathcal{C}_k$ typically depends on attributes or features of the data, such as categorical labels, ranges of numerical values, or other distinguishing characteristics relevant to the analysis. The result of $G(D)$ is a partitioning of the dataset D into disjoint subsets, where each subset corresponds to a unique value or category defined by $\mathcal{C}_k$. This operation is commonly utilized in tasks such as aggregation, summary statistics, or comparative analysis across defined groups.

Aggregation Operation A, which computes summary statistics or derived metrics for each group created by the group operation. The aggregation operation A can be expressed as:

$$A(G(D)) = \{a_k = f(S_k) : k \in K\},$$

where $S_k = \{x^{(i)} \in D \,|\, \mathcal{C}_k(x^{(i)})\}$, a_k represents the aggregated value for the k-th group, f is the aggregation function applied to the subset of data points in the k-th group, $k \in K$ iterates over all unique group identifiers, and the inner set $\{x^{(i)} \in D \,|\, \mathcal{C}_k(x^{(i)})\}$ defines the subset of data points belonging to the k-th group.

D Data Upload and Natural Language Query Interface

The workflow begins when a dataset (in CSV format) is uploaded by user. Once uploaded, the dataset undergoes a comprehensive pre-processing phase, including multiple steps, such as Data Validation, Missing Data Handling, and Exploratory Data Analysis (EDA), which are designed to ensure the data is clean, consistent, and ready for in-depth exploration:

The upload and initial analysis may take a few seconds, depending on the volume and complexity of the dataset. While this action might seem routine, it plays a vital role in ensuring the dataset is primed for analysis. A well-preprocessed dataset reduces the risk of errors, enhances the efficiency of downstream processes, and ultimately leads to more accurate and actionable insights. By automating these steps, the system minimizes the burden on the user, allowing them to focus on deriving value from the data rather than troubleshooting technical issues. This emphasis on robust pre-processing underscores the system's commitment to delivering reliable and high-quality results.

Upon successful upload and pre-processing of the dataset, the user is prompted to enter a query

in natural language. This step is crucial because it allows the user to define the specific insights they seek from the dataset, tailoring the analysis to their unique needs. The query input module is designed to be intuitive and flexible, supporting a wide range of queries related to Temporal, Spatial, and Attribute dimensions. By enabling users to ask questions in their own words, the system bridges the gap between complex data analysis and user-friendly interaction, making advanced analytics accessible to non-technical users. This user-driven approach ensures that the analysis remains focused and relevant, empowering users to uncover meaningful patterns, trends, and relationships within the data. The active engagement of the user through query input is not just a procedural step but a foundational aspect of the system's design, enabling personalized and actionable insights.

Example of Spatial-Temporal Heatmap Generation

This appendix elaborates on the heatmap visualization example, demonstrating how spatial-temporal filtering, grouping, and aggregation are applied to generate actionable insights.

Consider a dataset D containing spatial-temporal records, such as sensor measurements across geographic locations and timestamps. The visualization process begins with filtering F to isolate data within a specific spatial region (e.g., a city's administrative boundaries). Formally, $F(D) = \{x^{(i)} \in D \,|\, \mathcal{C}_{\text{spatial}}(x^{(i)})\}$, where $\mathcal{C}_{\text{spatial}}$ defines the geographic criteria.

Next, the grouping operation G organizes the filtered data into temporal intervals (e.g., hourly slots). Let $K_{\text{temporal}} = \{t_1, t_2, \ldots, t_n\}$ represent these intervals. The grouped data becomes $G(F(D)) = \{S'_k \,|\, k \in K_{\text{temporal}}\}$, where each S'_k corresponds to measurements within time interval k.

The aggregation operation A then computes summary statistics (e.g., mean measurement values) for each temporal group.

Finally, the visualization function V maps the aggregated data to a heatmap. The parameters Θ specify a color gradient (e.g., red for high values, blue for low) and spatial-temporal axes configurations. The resulting heatmap encodes spatial distributions of measurements across time intervals, enabling users to identify trends (e.g., peak pollution hours in specific neighborhoods).

This example illustrates how InsightMiner's pipeline transforms raw spatial-temporal data into interpretable visualizations, as formalized in the main text.

E More Discussion and Future Work

E.1 Hallucinations in MLLMs

The term "hallucinations" in the context LLMs refers to instances where these models generate text that is factually inaccurate, nonsensical, or ungrounded in reality. This phenomenon represents a well-documented limitation of LLMs, as their outputs often appear superficially plausible but lack logical coherence or factual validity.

Moreover, in VLMs, a distinct form of hallucination named value hallucination has been observed in existing implementations. For example, when generating descriptive annotations for figures, VLMs may produce text with incorrect numerical precision (e.g., mislabeling the y-axis scale in a histogram) despite demonstrating competent data-sorting capabilities. This specific issue may warrant further investigation in character-level VLM QA tasks.

A second type of hallucination observed in VLMs involves color intensity misinterpretation. When tasked with interpreting heatmaps designed to represent quantitative values, VLMs frequently struggle to distinguish between gradations of color depth. For instance, the model may inaccurately associate darker hues with higher values unless explicitly instructed to adhere to standardized color-value conventions. This limitation underscores the need for precise instructional input to guide VLMs in visual interpretation tasks.

E.2 Challenges in Structured Output Generation

When developing systems that rely on LLMs, it is essential to define specific protocols for backend processing. However, LLMs often struggle to produce outputs in a consistent, structured format. For instance, a system may require the output to be in JSON format with specific keys such as "code" and "operation." Yet, LLMs may fail to adhere to this requirement, either by omitting the "code" key or by generating outputs that deviate from the JSON format altogether.

Another challenge arises when the system attempts to process figures encoded in Base64 as inputs for VLMs. Although the underlying code

may be correct, errors can occur due to incorrect data inputs or the misuse of variable names for Base64-encoded figures. These inconsistencies can lead to processing failures or misinterpretation of the output, highlighting the need for robust error-handling mechanisms and stricter input validation protocols.

E.3 Limitations of VLMs with Interactive Visual Inputs

We observe a challenge in integrating VLMs with interactive figures lies in the limited support for dynamic input. Most VLMs are designed to process static images, making it difficult to incorporate interactive visual elements into the system. This constraint results in the loss of valuable information that might otherwise be accessible through interactive elements, such as tooltips, hover states, or dynamic visualizations.

To address this limitation, we have adapted the system to generate only static figures as inputs to the VLM. While this simplifies the process, it may lead to the exclusion of essential interactive features, diminishing the richness of the data available for interpretation. A potential solution is to extract relevant portions or snapshots from interactive figures, which can then serve as static inputs for the VLM.

Further experiments are needed to evaluate the effectiveness of this approach, exploring methods for aligning VLM inputs with interactive visual representations while maintaining the integrity of the information presented.

F Ai Assistants In Research and Writing

In this paper, we utilize AI assistants to enhance and polish the English writing by using prompts such as "Polish my English." Additionally, in our research, we design a novel interactive application based on AI assistants.