Veriscope: Provenance-Aware Verification of LLM Reasoning Artifacts in Conversational Data Analysis

Lingyu Meng
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, Zhejiang, China
kevinmeng@zju.edu.cn

Hanbei Zhan
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, Zhejiang, China
zhanhanbeiii@zju.edu.cn

Yuan Tian
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, Zhejiang, China
yuantian@zju.edu.cn

Yuchen He
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, Zhejiang, China
heyuchen@zju.edu.cn

Di Weng*
School of Software Technology
Zhejiang University
Ningbo, Zhejiang, China
dweng@zju.edu.cn

Yingcai Wu
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, Zhejiang, China
ycwu@zju.edu.cn

Abstract

Large language models (LLMs) are increasingly adopted by data analysts to perform various analytical tasks within conversational interfaces, owing to their capabilities in code generation and analytical reasoning. Conversational LLMs generate diverse artifacts through multiple reasoning steps and summarize analytical results into insights. Verifying such analysis remains essential yet challenging, as it requires examining both diverse artifacts and the underlying complex reasoning logic, which is not fully addressed by previous work. To facilitate validating LLM-generated results, we present a claim-centered view of analytic provenance and introduce Veriscope, an interactive system that supports fine-grained verification across interconnected artifacts, on-demand tracing of analytic provenance, and modification of analytical results. We formulated our design considerations based on findings derived from a formative study conducted with data analysts (N=6) who have prior experience in verification. We evaluated the effectiveness and usability of our method through a user study (N=18), which demonstrated that Veriscope substantially improves the efficiency of verifying analytical results produced by LLMs. The reliability of LLM-based features was assessed through a technical evaluation.

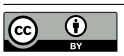
CCS Concepts

• **Human-centered computing** → **Interactive systems and tools**.

Keywords

Conversational Data Analysis, Insight Verification, Human-AI Interaction

*Di Weng is the corresponding author.



UIST '26, Detroit, MI, USA

ACM ISBN 979-8-4007-2856-3/26/11
https://doi.org/10.1145/3830398.3830599

ACM Reference Format:
Lingyu Meng, Hanbei Zhan, Yuan Tian, Yuchen He, Di Weng, and Yingcai Wu. 2026. Veriscope: Provenance-Aware Verification of LLM Reasoning Artifacts in Conversational Data Analysis. In *The 39th Annual ACM Symposium on User Interface Software and Technology (UIST '26), November 02–05, 2026, Detroit, MI, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3830398.3830599

1 Introduction

With the rapid development of large language models (LLMs) and their strong capabilities in reasoning over analytical contexts and generating code, analysts have increasingly turned to conversational LLMs such as Gemini and ChatGPT across a wide range of analytical scenarios [14, 18, 47]. When presented with datasets and user instructions, conversational LLMs engage in multiple steps of reasoning that interleave thinking, code generation, data processing, and visualization, before finally synthesizing findings into a summarized answer. This process involves a variety of objects including data tables, code, execution results, charts, and insights, which we collectively refer to as *reasoning artifacts* in this study. While these artifacts can provide useful analytical results, their validity can be affected by numerous factors such as misalignment with user intents [49], misunderstanding of analytical contexts [3], hallucinations in code generation [43, 50], and misinterpretation of visualizations [17, 41]. These risks make it critical for end users to verify LLM-generated outputs.

However, verifying the analytical results produced by conversational LLMs remains a challenging task, as it demands not only careful examination of diverse artifacts, but also a coherent understanding of the analytic provenance [25, 32] that connects them. For instance, validating a single insight requires users to first locate relevant artifacts, such as the charts on which the insight was based and the code that produced it, and assess their mutual consistency. Users may then need to trace further back through prior reasoning steps to verify that the underlying data transformations were correct and that no errors were introduced earlier in the process. In current conversational interfaces, however, these artifacts are scattered across lengthy chat histories, forcing users to manually scroll, search, and mentally reconstruct the reasoning chain, which is inherently tedious and error-prone.

Prior research has explored various approaches to help users better navigate and understand LLM-generated content. For instance, Graphologue [20] and Sensecape [36] convert linear LLM responses into interactive visual structures to support exploration and sensemaking, but prioritize helping users comprehend textual outputs over verifying analytical reasoning. In the context of LLM-assisted data analysis, Gu et al. [14] and InsightLens [44] help analysts navigate and organize analytical conversations by structuring them into identifiable elements such as topics, analytical components, and insights, but focus on supporting comprehension and communication rather than systematic verification. To specifically address verification, several studies have proposed tools that surface analytical artifacts for inspection, such as code visualizations [47, 54], editable execution plans [22], and multi-modal representations pairing code with explanations, tables, and charts [15]. However, these approaches either primarily focus on a single type of artifact or support inspection of individual analysis steps in isolation. They do not establish fine-grained connections across diverse artifacts, for example, linking specific phrases in an insight to corresponding elements in a chart and the code that produced it, nor do they enable users to extract the end-to-end analytic provenance that captures how a particular finding was derived through a chain of preceding reasoning steps.

Motivated by these limitations, we aim to support the verification of LLM-generated analytical results within conversational interfaces by enabling users to navigate interconnected reasoning artifacts and validate the underlying reasoning process. To achieve this, we present a claim-centered view of analytic provenance, which refers to the coherent chain of diverse artifacts across prior reasoning steps that collectively led to one particular claim, a verifiable statement within summarized insights. To characterize the workflows and challenges of such verification, we also conducted interviews with six data analysts who were familiar with LLM-assisted data analysis and had experience verifying the resulting outputs in conversational interfaces, and formulated five design considerations based on findings from the interviews.

Guided by these considerations and the claim-centered perspective, we developed **Veriscope**, a prototype system that supports systematic verification of LLM responses through tailored interaction workflow. Veriscope establishes fine-grained connections across reasoning artifacts, enabling efficient navigation and intuitive cross-validation among insights, charts, code, and data tables. Given one specific insight, the system can also automatically extract the analytic provenance from preceding steps and present a step-by-step summary of reasoning logic alongside corresponding artifacts, allowing users to trace how a particular finding within summarized insights was derived. In addition, the system supports both direct editing and guided re-generation to facilitate correction of identified errors.

To evaluate the effectiveness and usability of Veriscope, we conducted a user study (N=18) comparing it against a baseline system. Results showed that our approach substantially improves verification efficiency, reduces errors, and enhances user experience in validating LLM-generated analytical results. We also conducted a technical evaluation to assess the reliability of LLM-based features including provenance extraction and keyword mapping.

Our contributions can be summarized as follows:

- We conducted a formative study (N=6) that characterized the workflows and challenges of verifying LLM-generated analytical results and formulated five design considerations.
- We propose a claim-centered view of analytic provenance and interaction workflow for verifying LLM-generated analysis, realized in Veriscope, a prototype system that supports provenance-aware verification through fine-grained cross-artifact linking, automatic provenance extraction, and error correction.
- We conducted a user study (N=18) demonstrating the effectiveness and usability of the proposed approach, and a technical evaluation assessing the reliability of LLM-based features.

2 Related Work

In this section, we discuss related work of our study in three aspects: LLMs for data analysis, exploring and verifying LLM responses, and tracing and visualizing analytic provenance.

2.1 LLMs for Data Analysis

A growing body of research works has integrated large language models (LLMs) into data analysis workflows for various purposes, such as data wrangling [39, 40], visualization authoring [37, 39, 40], visual analytics [51–53], and data exploration [18, 38, 44]. By leveraging the natural language understanding and code generation capabilities of LLMs, these integrations enable users to execute various analytical operations.

In addition, as foundational models become increasingly capable in their logical reasoning and end-to-end data analysis capabilities, general-purpose conversational interfaces like ChatGPT and Copilot are favored by analysts for the tasks of data processing, insight discovery, and visualization. Building upon the widespread adoption of these interfaces, several recent studies have proposed augmented systems designed to further enhance the interactive data analysis experience and address the limitations of purely chat-based platforms [5, 44, 47]. For example, Weng et al. [44] developed InsightLens, an interactive system which collectively visualizes diverse insights generated by LLM-assisted analysis, alongside their analytical contexts. Chen et al. [5] introduced multimodal interactions into chat interfaces, enhancing user intent communication during visual analysis of datasets.

2.2 Exploring and Verifying LLM Responses

With LLMs' growing capabilities in generating answers for diverse prompts, it is increasingly necessary for end-users to explore and verify the resulting outputs before applying them to downstream tasks. A number of research works studied interactive approaches to explore and evaluate multiple LLM responses from different samples [13], models [1, 21], or user prompts [1]. Another line of research utilized novel visualization and interaction techniques to facilitate the exploration and sensemaking of LLM responses in various scenarios. Graphologue [20] transformed entities and relations within the generated text into node-link diagrams for flexible information seeking and comprehension. Other systems like Sensecape [36] and HIPPO [30] leveraged multi-level layouts for users to hierarchically explore responses at different information and reasoning granularities. More recently, Gu et al. [14] conducted

an empirical study, investigating the process of analysts navigating and understanding multi-turn conversations.

In addition to exploration, the verification of LLM responses is critical to ensuring the integrity of analytical conclusions. Interactive systems such as HILL [23] and RELIC [7] were developed to help users validate the truthfulness of the generated answers and mitigate potential hallucinations. Meanwhile, Gu et al. [15] specifically investigated patterns of verification workflow for results and artifacts from LLM-assisted data analysis, and summarized implications for future tool design. Interactive tools have been developed to assist the verification of LLM-assisted data analysis in different aspects. For verifying LLM-generated code, WaitGPT [47] utilized a side diagram to simultaneously visualize data operations during code generation, supporting efficient verification and steering of the analytical process. ViseGPT [54] combined unit test and interactive Gantt chart to evaluate the correctness of data wrangling scripts and provide visual feedback for iterative refinement. For verifying and steering LLM-assisted data analysis process, Kazemitabaar et al. [22] proposed editable interfaces for users to modify task decomposition at phasewise and stepwise granularities.

Compared with previous research, our study targets multi-step reasoning of LLMs, aiming to conduct validation on both various interconnected reasoning artifacts produced during the analytical process, and the underlying analytic provenance to help users identify potential errors and modify analytical results.

2.3 Tracing and Visualizing Analytic Provenance

There are many studies in the community of visualization and HCI conducted on tracing and visualizing provenance in the process of data analysis. These studies focused on provenance in various analytical environments or scenarios, such as visual analytics system [26, 27, 33], computational notebooks [11, 12, 45], workflow system [4], dashboard [35], and provenance-specific tools [28]. For example, the most typical one is VisTrails [4], which facilitated the efficient recovery of scientific insights by maintaining records of both dataflows and visualizations. Sarvghad et al. [33] incorporated the historical frequency of analyzing data dimensions into a visual analytics system to encourage the discovery of new insights. Stitz et al. [35] developed KnowledgePearls, which leveraged a partial matching mechanism to retrieve previous analytical results.

Unlike existing research which records human-authored workflows for recall and reproducibility, Veriscope targets the reasoning artifacts LLMs produce autonomously during analysis and helps users verify specific claims within generated insights against them. This raises two challenges absent from logging-based systems. First, derivation links are implicit, because LLMs emit thoughts, code, results, charts, and insights as an undifferentiated sequence, without recording which of these support any given claim. Second, provenance is claim-specific, since different sentences depend on different subsets of prior steps, so its scope cannot be precomputed. Our prototype system can systematically extract a coherent chain of diverse artifacts leading to selected claims within insights. Different from prior LLM tools (e.g., WaitGPT [47]), which inspect data operations in each analytical step, Veriscope validates if a claim stays consistent with the extracted chain that produces it.

3 Background and Problem Formulation

As the capabilities of LLMs continue to advance, LLMs have demonstrated the ability to perform multi-step reasoning over complex scenarios [6]. We define the key concepts and error types that arise during LLM-assisted data analysis as follows.

Modern LLMs with multimodal capabilities like Gemini typically follow an iterative reasoning process [24, 42]. In the context of data analysis, given an analytical task and a dataset, the model first plans its approach during the thinking process, then generates code to perform data processing and construct visualizations. The code is executed by an external runtime to produce textual execution results and charts. The model then analyzes these outputs, draws intermediate conclusions, and plans further analytical actions based on its findings. This cycle repeats across multiple reasoning steps. Once the full process concludes, the model synthesizes the intermediate conclusions into a final set of summarized insights. We refer to the diverse outputs produced throughout this process as *reasoning artifacts*, which comprise six categories: the original data table, thoughts, code, execution results, charts, and the final summarized insights. These artifacts span the full analytical process and vary in modality, including text, code, tabular data, and visualizations. Based on the above process, we propose a claim-centered view of analytic provenance. Specifically, a *claim* is a verifiable statement within an insight, and *analytic provenance* refers to the coherent chain of artifacts across reasoning steps that collectively leads to one particular claim (Figure 1). A claim does not possess a fixed granularity. It may span a single clause, encompass multiple sentences, or constitute an entire paragraph of insight. Claims are not segmented in advance. Rather, their boundaries are dynamically determined by the user's selection. Meanwhile, since the reasoning process is inherently iterative and distinct findings may draw upon different subsets of prior steps, the scope and composition of the provenance chain naturally vary depending on the specific claim being examined.

Across the iterative reasoning process, we identify four categories of errors that arise at the transitions between artifact types: from thoughts to code, from code to execution results and charts, from execution results and charts back to subsequent thoughts, and from intermediate thoughts to the final summarized insights. Among these, the transition from code to execution results and charts is typically reliable, as code execution is delegated to an external runtime rather than performed by the model itself. The remaining three transitions correspond to three primary error types supported by the literature review:

- **Incorrect code generation** (thoughts → code): the model's thinking identifies an analytical intent, but the generated code selects incorrect data columns, applies flawed operations, or otherwise misimplements the intended logic [43, 47, 50, 54].
- **Misinterpretation of charts and execution results** (execution results and charts → thoughts): the model misreads visual patterns, overlooks data details in execution outputs, or draws incorrect conclusions when reasoning about these artifacts in the next step [17, 29, 41, 46].
- **Inaccurate summarization of insights** (thoughts → insights): when synthesizing intermediate findings across reasoning steps, the model could introduce hallucinations, overgeneralize from

Figure 1: LLMs implement multi-step reasoning to synthesize insights. The analytic provenance of one claim with the insights is a coherent chain of artifacts across reasoning steps.

context-specific findings, or remove important qualifications [2, 31, 48], leading to incorrect analytical conclusions.

These three error types cover all transitions where the model actively generates or interprets content, and each can either invalidate individual summarized insights or cascade through the reasoning chain, motivating the need for provenance-aware verification.

Taken together, the problem addressed in this work is: given the reasoning artifacts produced during LLM-assisted data analysis, how can we enable analysts to efficiently verify the correctness of summarized insights, trace the analytic provenance underlying specific findings, and identify and correct errors that arise across the reasoning process? To inform our approach, we first conducted a preliminary study, detailed in the next section, to understand how analysts currently verify LLM-generated analytical results and what challenges they encounter in practice.

4 Formative Study

In this section, we introduce our formative study[1] to characterize the workflow and challenges of verifying LLM-assisted data analysis, and then summarize our findings and design considerations.

4.1 Interviewees and Procedure

We recruited six data analysts (P1–P6, 3 female, 3 male, aged 20–30). All participants were doctoral students working on interdisciplinary projects involving data analysis across domains such as AI4VIS, exploratory data analysis, sports analytics and time series analysis. Each participant had at least two years of experience using LLMs for data analysis, with varying levels of proficiency in verifying LLM-generated results.

During the interviews, participants first walked through their own past analysis sessions, describing the datasets and prompts they had provided to LLMs along with the corresponding analytical outputs. They then elaborated on their general workflows for verifying these outputs, including the sequence of examination steps, the artifacts they prioritized, and the challenges they encountered throughout the process. Each interview lasted approximately 40 minutes and was transcribed for analysis. We analyzed the interview data using thematic analysis [8]. Two researchers independently coded the transcripts, then discussed and consolidated themes through iterative review.

4.2 Findings

Our interviews revealed four key findings regarding how analysts verify LLM-generated analytical outputs, what challenges they face, and what support they need.

F1: Verification workflows and depth varied across analysts. Participants adopted different strategies for initiating verification. Half (3/6) began by reviewing insights, then searched for supporting artifacts. For example, P6 appreciated the structured insight summaries and bold keywords produced by LLMs, noting that they helped her quickly grasp the key points and approach subsequent verification with clear focus. The other half (3/6) started from charts to gain an overview of the analytical process. Participants also differed in verification depth. Some, like P4, focused primarily on surface-level errors: "*I would conduct a more thorough inspection only if the charts show obvious errors. Otherwise, I tend to trust the LLM's overall analytical results.*" Others (3/6) rigorously verified the consistency between textual insights and charts, ensuring the accuracy of outputs across artifact types. P1 also mentioned that she would sometimes trace back to data tables to examine whether the numbers within insights were correct.

F2: Reasoning artifacts are essential for verification but difficult to examine. During the analytical process, LLMs generate various types of reasoning artifacts, including thoughts, code, execution results, and charts. Participants found each type useful for verifying different aspects of the analysis. For instance, participants used code for checking data transformation correctness and charts for validating the insight text. In particular, most participants (5/6) prioritized charts as the key evidence for verification, since they are directly tied to insights and easier to interpret visually. However, all participants (6/6) expressed frustration with the manual effort required to cross-reference information across multiple artifacts. P6 noted: "*Before I really get down to verifying one specific insight, it takes me much effort to identify relevant code or chart. Sometimes I may even choose the wrong one for validation.*" P5 expressed a similar concern: "*Even though I really want to have a detailed examination, the sheer volume of text and code puts me under pressure and makes me hesitate.*" Without appropriate guidance, manual examination left participants struggling to locate key information across artifacts and at risk of overlooking important details.

F3: Analytic provenance is important for validating the whole reasoning process. Although examining individual artifacts provides straightforward verification, analytic provenance,

[1]The study has been approved by State Key Lab of CAD&CG, Zhejiang University.

which is a series of analytical operations and artifacts that lead to a given insight or finding, captures the broader reasoning chain that participants (3/6) considered necessary for thorough validation. P1 noted: "*Sometimes I find that even though the current insight may be correct with corresponding visualization, it turns out to be irrelevant and derived from an initially incorrect finding when I trace back through previous analytical steps.*" P5 echoed this concern: "*Only by reviewing all the critical artifacts along the reasoning path can I ensure that the LLM's behavior is on the right track.*" P5 also noted that analytic provenance could facilitate cohesive verification across multiple insights without laboriously checking individually.

F4: Participants required different levels of correction depending on issue severity. When only minor inaccuracies appeared in specific details of the insights, participants (5/6) typically opted to directly modify the outputs. P1 pointed out: "*If I ask LLMs to regenerate a lot of content, I'm worried it might mess with the parts I've already confirmed, so just directly editing it is the simplest and most efficient way to go.*" When major flaws were identified in the analytical procedures or the overall reasoning process, participants (6/6) would instead flag the errors to LLMs and either regenerate incorrect parts or restart the analysis entirely.

4.3 Design Considerations

Based on the above findings, we formulated five design considerations to inform the design of our prototype system.

DC1: Highlight verifiable elements and associate them with corresponding artifacts (F1, F2). To help users comprehend LLM-generated insights and identify what can be validated, the system should identify and highlight verifiable elements within analytical insights that correspond to specific data fields or visual elements. Furthermore, the system should establish explicit associations between insights and their corresponding artifacts, such as charts, enabling users to quickly locate relevant evidence and initiate the verification process.

DC2: Visualize diverse reasoning artifacts with summarized overviews (F1, F2). The system should present a structured visualization of various reasoning artifacts generated during the LLM analytical process, including intermediate thoughts, code, execution results, and charts. The visualization should support both overview comprehension and detailed inspection. Beyond presenting the detailed content of each artifact, the system should distill key findings and procedures into concise summaries, enabling users to rapidly comprehend the overall reasoning process without being overwhelmed by dense text and code.

DC3: Enable interactive cross-validation across associated artifacts (F2). Leveraging associations between insights and artifacts, the system should support fine-grained, in-place cross-validation among correlated artifacts without requiring users to manually locate corresponding content. For instance, the system should allow users to verify whether specific keywords within insights accurately align with visual elements in associated charts, trace which code snippets produce a given chart, and query data tables to validate statistics referenced in insights.

DC4: Support on-demand tracing of analytic provenance (F3). Upon selection of a specific claim within insights, the system should identify and extract relevant artifacts from prior reasoning steps. The extracted artifacts should be organized as a coherent reasoning chain, accompanied by a step-by-step summary of the underlying reasoning logic to facilitate comprehension. The system should also allow users to filter provenance by artifact type, aligning with established provenance taxonomies [32], and to navigate to specific artifacts for detailed examination.

DC5: Enable modification and correction of analytical results (F4). The system should support two levels of correction to address issues of varying severity. For minor inaccuracies, users should be able to directly edit insight text or code and re-execute to obtain updated results. For major flaws in the reasoning process, users should be able to reference specific artifacts and describe identified issues to instruct the LLM to revise the relevant analytical steps accordingly. Throughout both processes, the system should maintain editing history, ensuring all modifications are traceable.

5 Veriscope

Based on the design considerations, we designed and developed Veriscope, a prototype system that facilitates provenance-aware verification of LLM-assisted data analysis. Our system comprises two views: the **chat view** (Figure 2a), where users enter prompts and review analytical insights summarized from the reasoning process, and the **analysis view** (Figure 2c), which visualizes the diverse artifacts generated during the LLM analytical process.

In this section, we first illustrate the basic workflow of Veriscope through a usage scenario, and then explain Veriscope's core features, namely, enhancing LLM responses, visualizing diverse reasoning artifacts, cross-validating artifacts, extracting and tracing analytic provenance, and correcting analytical results.

5.1 Usage Scenario

Alex is a senior business analyst at a retail company, tasked with analyzing historical sales data in order to optimize operational strategies. To ensure the accuracy and reliability of his analysis, he chose to use Veriscope for data analysis and further verification. He uploaded the dataset and entered his prompts. Then Veriscope generated a series of artifacts during the thinking process in the analysis view (**DC2**) (Figure 2c), and presented summarized insights in the chat view (Figure 2a).

Alex began by examining the first insight. He first opened up the correlated chart (**DC1**), and hovered over underlined keywords to highlight associated data points in the visualization (**DC3**), finding that between sales and profit, only the monthly trend of sales aligned with the insight's description. He therefore planned to remove "profit" from the description.

Afterwards, Alex wanted to verify insights related to *Profit*, *Category*, and *Sub-Category*, so he clicked on the corresponding field names at the top of the insights and located the second and third insights (**DC1**). Starting from the second one (Figure 2b), he continued to leverage underlined keywords to validate observations from the chart (**DC3**). Alex soon discovered that the chart did not display the feature described in the last sentence, as Office Supplies did not show a noticeable loss (Figure 2e). So he brushed over this sentence to extract relevant artifacts in the reasoning process that led to this conclusion (**DC4**). Veriscope subsequently presented extracted artifacts and offered a step-by-step summary of the underlying logic in

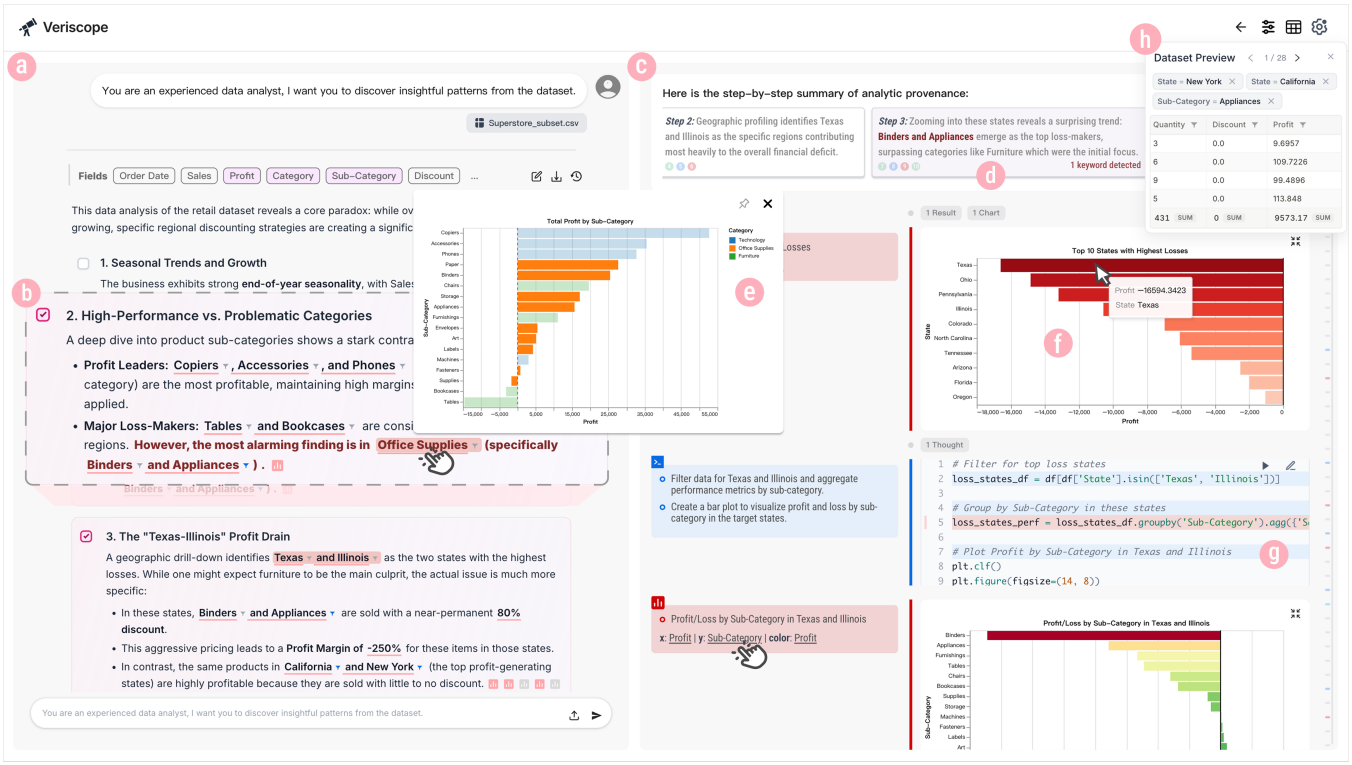


Figure 2: The user interface of Veriscope. (a) The chat view presents the summarized insights. (b) Each insight is enhanced with underlined keywords and associated with relevant artifacts. (c) The analysis view visualizes artifacts generated during the analytical process. (d) The extraction of analytic provenance and its step-by-step summary. (e-f) Bidirectional highlighting between keywords and visual elements. (g) Tracing code snippets via chart encoding. (h) Keyword-driven table queries.

the analysis view (**DC4**). After reviewing the summary (Figure 2d), Alex made a preliminary judgment that this finding was drawn from analyzing two of the most underperforming states, Texas and Illinois, rather than reflecting the overall situation. He then proceeded to filter out thoughts and navigate to specific artifacts for more detailed examination (**DC4**). During the review process, Alex found that most of the artifacts were generated in the context of analyzing Texas and Illinois, and several charts were related to the third insight (**DC1**). He hovered over these charts (Figure 2f) to check whether the key patterns and statistics were mentioned in the insight (**DC3**), and also clicked on the chart encodings (Figure 2g) to verify corresponding operations in the code (**DC3**). Ultimately, Alex confirmed his initial assessment and decided to move the identified sentence to the third insight.

Having gained a deeper understanding of the overall process, Alex continued validating the remaining insights. Notably, one chart showed that the discount for Appliances in California and New York was zero. Alex was skeptical of this, so he clicked on the filter icons next to the associated keywords to filter the data table (**DC3**), and found that the sum of the discount values was indeed zero (Figure 2h), which confirmed this conclusion. Finally, Alex clicked on the edit icon to launch the Markdown editor, revising insights based on previously identified errors, optimizing certain phrasing, and exporting the final results (**DC5**).

5.2 Enhancing LLM Responses

Veriscope presents user prompts and summarized insights in the chat view (Figure 2a). To help users understand and efficiently locate information relevant to data patterns within analytical insights, Veriscope first extracts critical keywords for visual enhancement. The system first re-executes the LLM-generated code to retrieve the schema of each chart's intermediate table, including field names and their associated values or value ranges, alongside the visual encoding channels used to render each field. Given the chart information and corresponding table schemas, the system then leverages LLMs to extract keywords that can be explicitly or implicitly mapped to specific fields of one particular table schema, based on the context of the corresponding sentences (Figure 3a). These keywords are subsequently transformed internally into values that conform to the required field formats (Figure 3b). After keyword extraction, each keyword is underlined in the interface and associated with one chart within the LLM reasoning process. Users can click these keywords to open the corresponding charts right next to them.

To provide users with analytical context and direct evidence for validation, Veriscope builds direct connections among critical artifacts. The system leverages LLMs to tag each insight with relevant data fields, enabling users to select field names to locate specific insights. The system also identifies all charts associated with the keywords present in each insight, and displays them as icons at the

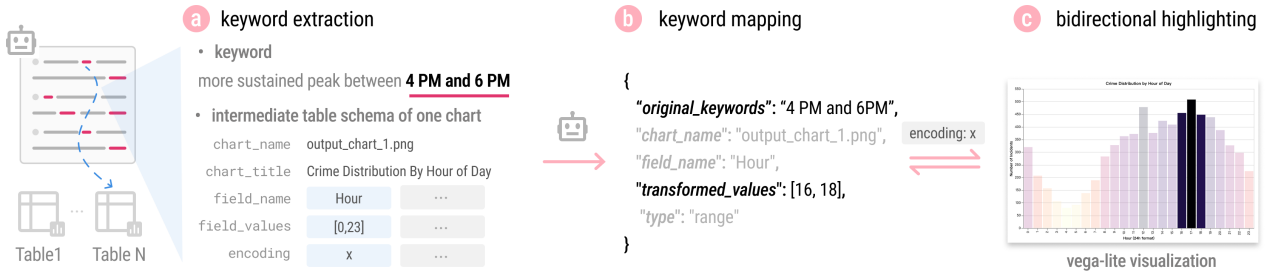


Figure 3: (a) Veriscope extracts keywords mappable to the intermediate table schema of one chart. (b) Extracted keywords are transformed to values that conform to the schema. (c) Bidirectional highlighting is supported with keyword mappings.

end of insights. Users can hover over these icons for quick preview or click to navigate directly to their corresponding positions in the analysis view (Figure 2c). Conversely, hovering over charts in the analysis view highlights the associated insights in the chat view.

5.3 Visualizing Diverse Reasoning Artifacts

Veriscope visualizes diverse reasoning artifacts generated throughout the analytical process in the analysis view (Figure 2c), including *thought*, *code*, *execution result*, and *chart*. Users can also click the table icon to inspect the original *data table*. To assist users in understanding artifacts with dense textual content, Veriscope distills key findings or procedures from each artifact, and displays the summary alongside detailed information. For *chart*, Veriscope leverages its encoding information extracted from code to present its basic attributes. To facilitate examination of artifacts, the system allows users to click on one specific key point to jump directly to the corresponding content. The navigation sidebar also visualizes the distribution of diverse artifacts along the reasoning process. Each item on the sidebar supports artifact preview and quick navigation to the relevant location. Additionally, the system leverages LLMs to transform generated Python code into Vega-Lite [34] specifications, and renders them as interactive visualizations, where users can hover over visual elements to inspect specific data values and retrieve information valuable to verification (Figure 2f).

5.4 Cross-Validating Artifacts

Building upon the connections among critical artifacts, Veriscope further enhances the user experience by providing three intuitive interactions for fine-grained verification.

Bidirectional highlighting between keywords and visual elements. As demonstrated in the usage scenario (Section 5.1), this interaction effectively facilitates detailed validation of the consistency between keywords in the insights and visual elements within the charts. On the one hand, when users hover over keywords, the system automatically utilizes their internally transformed values to modify the corresponding Vega-Lite specification via alignment of visual encodings, emphasizing associated visual elements while de-emphasizing the remaining ones through reduced opacity (Figure 2e). On the other hand, users can also hover over visual elements, which in turn highlights keywords in the insights with matched values (Figure 2f).

Tracing code snippets via chart encoding. To facilitate more granular verification of the code responsible for one specific chart, the system allows users to click on the visual encoding information shown in the chart summary, leveraging keyword search to locate directly related code snippets (Figure 2g), enabling users to inspect and validate if appropriate data operations, field names, and parameters are used for transforming data and creating visualizations.

Keyword-driven table queries. Querying over data tables is the most direct approach to validate analytical insights and statistics shown in the charts. Therefore, our system specifically annotates underlined categorical keywords whose texts or transformed values appear in the data tables, and enables users to perform automatic filtering on the original data tables by clicking the icons beside specific text (Figure 2h). Users can further perform aggregation over specific fields in the table to extract key numeric values for comparison against charts and insights. Meanwhile, we expect future work to integrate more advanced features to support flexible queries over both original and intermediate tables.

5.5 Extracting and Tracing Analytic Provenance

To enable users to efficiently extract analytic provenance and trace corresponding reasoning process, our system integrates LLMs to first coherently retrieve the contents of relevant artifacts, and provide a systematic summary of the underlying reasoning logic.

To start with, users can brush over insight text in the chat view. The system then instructs the LLM to review artifacts from prior analytical procedures and extract a reasoning chain comprising a series of intermediate findings, essential data transformations, critical execution results and visualizations that collectively lead to the selected content. After the extraction, sub-contents of relevant artifacts are further highlighted while irrelevant artifacts are folded (Figure 2c). Users can also filter the selected contents by the type of artifacts, exploring different categories of analytic provenance mentioned in the taxonomy proposed by Ragan et al. [32]. Meanwhile, users may still find it challenging to interpret the extracted provenance if it involves a large number of reasoning steps. To address this issue, Veriscope further leverages LLMs to decompose and summarize the reasoning logic in a step-by-step manner, with underlined keywords within the selected content additionally highlighted for users to locate critical steps. Each step is also accompanied by referenced artifacts as supporting evidence. For each artifact, users can hover over an associated indicator for preview or click to navigate to its location for detailed examination (Figure 2d).

Rather than pre-computing provenance for all summarized insights, Veriscope adopts an on-demand strategy. Extracting analytic lineage is computationally expensive, consuming substantial tokens and introducing latency that would degrade conversational responsiveness. Moreover, the granularity at which an analyst wishes to verify an insight cannot be determined in advance: one may want to trace an entire paragraph-level summary, while another may question several sentences or a claim within one sentence. On-demand tracing addresses both issues by deferring extraction until the analyst selects the claim to verify, reducing unnecessary computation while adapting to the chosen level of detail.

5.6 Correcting Analytical Results

Veriscope provides two primary ways for users to make corrections. First, users can click on the edit button in the chat view to launch a Markdown editor, where they can directly modify the LLM answer. The system also supports directly editing the incorrect code snippets and re-execution to obtain updated results. Second, our system supports guided regeneration of analytical results. Specifically, users can describe the identified issue in the input field, referencing specific artifacts with at-sign commands or brushing interactions, and specify either partial or full regeneration. The system will then re-invoke the LLM to revise and regenerate the relevant artifacts accordingly. The system also maintains a complete editing history, helping users keep track of modifications.

6 Implementation

We implemented Veriscope as a web-based prototype system consisting of a frontend interface and a backend server. The frontend is built with Vue.js and Vuex, while the backend is developed with Flask and Python, communicating with the frontend via RESTful API. Veriscope leverages `gemini-3-flash-preview` API for the design features described in Section 5, owing to its low latency and multimodal understanding capabilities.

7 User Study

We conducted a user study[2] with 18 participants to evaluate the effectiveness and usability of Veriscope.

7.1 Study Setup

7.1.1 Participants. We recruited 18 participants (P1–P18, 11 males, 7 females, age 19–28) from diverse backgrounds, including computer science, mechanical engineering, business administration, etc. All participants had prior experience in using conversational LLMs for data analysis and were familiar with the artifacts produced in this process. Participants received RMB 60 per hour as compensation.

7.1.2 Baseline. We adopted a baseline system with the same layout and visual styling as Veriscope, presenting summarized insights in the left panel and displaying artifacts produced during the reasoning process in the right panel. Meanwhile, it omitted design features for direct artifact association, fine-grained verification, analytic provenance extraction, and error correction, while retaining basic support for artifact-type filtering and output regeneration. Both systems were evaluated under identical hardware conditions. This isolated the contribution of mechanisms without confounding factors including interface layout, artifact availability, and model behavior. Commercial tools such as ChatGPT and Claude were not selected as baselines, as they address a complementary question, real-world utility, and cannot attribute gains to specific mechanisms without a controlled environment.

7.1.3 Tasks. We designed four tasks (Tasks 1–4) to evaluate both Veriscope and the baseline system, each grounded in LLM-generated outputs derived from analyzing four distinct datasets, including Car Sales Report[3], Disney+ Movies and TV Shows[4], JHU CSSE COVID-19 Data[5] [10], and Superstore Dataset[6]. For the first, third, and fourth datasets, we used a subset of the data for this evaluation. In Task 1, participants were asked to extract the analytic provenance of an assigned insight and articulate the underlying reasoning process of LLMs. In Tasks 2–4, without being informed of the error types in advance, participants were asked to identify a distinct category of errors related to assigned insights within each output (summarized in Section 3) and modify the generated results accordingly. Participants were required to use either Veriscope or the baseline system to complete each task. Tasks were defined around verification outcomes, including understanding provenance and detecting the three error types in Section 3, rather than specific Veriscope interactions.

7.1.4 Procedure. We adopted a counterbalanced procedure similar to that of ViseGPT [54], which divided 18 participants into 6 groups. Each group consisted of 3 participants who worked on 4 tasks (Task 1–4) in one distinct sequence of using Veriscope and the baseline system. The assignment of systems to tasks was rotated across groups following a modified balanced Latin square scheme [9], ensuring that each system was used for each task by an equal number of 9 participants.

During the study process, we first introduced the general background and the design features of Veriscope to participants. After the tutorial, participants were allowed to freely practice the functionalities of Veriscope for about 5 minutes. Prior to each task, participants received a brief overview of the associated dataset, including its background, data attributes, and the specific meaning of each column. Participants were then asked to complete each task following their pre-assigned system sequence, with a time limit of 10 minutes per task. Completion time excludes the system's preprocessing of reasoning artifacts and is later analyzed for successful completion. Upon the completion or failure of one task, participants were required to fill in a modified NASA-TLX questionnaire [16] with a 7-point Likert scale. Lastly, we conducted semi-structured interviews with participants to collect their feedback. The entire process lasted about 60–90 minutes and was recorded for analysis.

7.2 Quantitative Results

7.2.1 Success rates and completion time. Table 1 and Figure 4 show that Veriscope performs better than the baseline system in success rates and completion time across Task 1–4. To complete Task 1 and Task 4, participants have to extract the corresponding reasoning

[2]The study has been approved by State Key Lab of CAD&CG, Zhejiang University.

[3]https://www.kaggle.com/datasets/missionjee/car-sales-report
[4]https://www.kaggle.com/datasets/shivamb/disney-movies-and-tv-shows
[5]https://github.com/CSSEGISandData/COVID-19
[6]https://www.kaggle.com/datasets/vivek468/superstore-dataset-final

Table 1: Task success rates (%) of Veriscope and Baseline. The purposes were not revealed to the participants.

Task	Purpose	Success (%)	
		Veriscope	Baseline
1	understand analytic provenance	100	66.67
2	identify errors in chart interpretation	77.78	44.44
3	identify errors in code generation	77.78	55.56
4	identify errors in insight summary	88.89	55.56

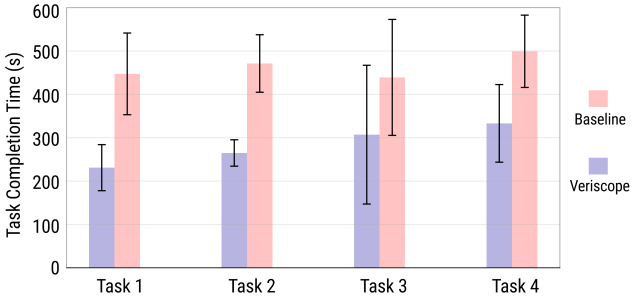


Figure 4: The task completion time (mean ± SD) for the baseline system and Veriscope.

chain of LLMs. This process is found to be more effective and much more efficient with Veriscope compared to the baseline system. In Task 4, P7 and P13 either spent considerably more time or failed the task. In fact, they successfully extracted the reasoning chains earlier during the task, but they misread or skipped certain steps in the step-by-step summary, which led to their poorer performance. In Task 2, participants achieved high efficiency and steady performance through bidirectional highlighting between underlined keywords and associated visual elements in Veriscope. Several participants failed the task because they only skimmed through the highlighted patterns or had difficulty interpreting certain terms for describing data attributes. The gap between Veriscope and the baseline system is relatively smaller in Task 3. With Veriscope, participants presented a high variance of completion time. Participants were generally able to locate relevant charts and code snippets quickly. However, those less familiar with tabular datasets and data wrangling still found it challenging to identify incorrect data operations within the code. This highlights an opportunity for future improvement to help users better understand code operations.

7.2.2 Ratings. Based on the result of non-parametric Wilcoxon signed-rank tests, the ratings of Veriscope are significantly better than the baseline system across all of the measured dimensions in the NASA-TLX questionnaire ($p<.001$). The result suggests that Veriscope substantially improves the user experience of verifying LLM-assisted data analysis by greatly lowering both cognitive and physical burden, and increasing satisfaction level during the verification process (Figure 5).

7.3 Qualitative Results

In this section, we report the qualitative findings derived from participant behavior observed during task execution and the feedback we received from semi-structured interviews.

7.3.1 Feedback on interactive features. Most participants (16/18) praised the direct connection between insights and charts, through which they immediately locate the most relevant evidence alongside its context, allowing for further verification. Meanwhile, participants were impressed with the bidirectional highlighting between keywords and visual elements, describing it as *magical* and *intuitive*. P4 commented: "*Those underlined keywords help me quickly pull out the key information and the highlighting also let me jump straight to the relevant part of the charts, instantly grasping the visual pattern. The whole process is simply effortless.*" P6 also appreciated the interaction of using the chart encoding to highlight associated code snippets, noting: "*This feature is pretty useful for me to locate critical data operations, especially when the volume of code gets big.*"

7.3.2 Feedback on extracting analytic provenance. 14 participants out of 18 complimented the extraction of analytic provenance and acknowledged that it played a critical role in the verification process. P15 pointed out: "*The extracted reasoning chain can help me construct a verification framework, enabling me to subsequently conduct structured exploration.*" P16 also mentioned that extracted provenance gave her a "*global perspective*", whereas she could be stuck in localized analysis steps while overlooking other parts with the baseline system. Participants also found the step-by-step summary of analytic provenance useful. P14 particularly liked the feature of presenting referenced artifacts within each reasoning step, allowing her to quickly navigate to the corresponding analytical context for targeted verification. Meanwhile, P14 suggested integrating more graphical representations to help users better understand the underlying reasoning logic. P7 recognized the effectiveness of step-by-step summary of the provenance, but also proposed: "*It would be even better if the system can directly tell me whether the selected answer is aligned with the reasoning process.*"

7.3.3 Verification workflow with Veriscope. While using the baseline system, participants generally did not opt to first comprehend the reasoning logic of LLMs. They instead directed their efforts from the start toward identifying the artifacts directly relevant to the assigned insights. With Veriscope, we found that several participants would, at early stages of the verification process, brush over insights to extract the corresponding analytic provenance, and subsequently conduct a more fine-grained verification during or after this extraction process. When examining the consistency between insights and charts, participants using the baseline system would typically read through an insight to understand its meaning, locate the relevant chart, and then revisit the entire insight again to build connections. In Veriscope, by contrast, participants directly clicked and hovered on underlined keywords to highlight the corresponding visual patterns in the chart, leveraging these linked visual cues to cross-reference the insights without repeated back-and-forth examination.

7.3.4 Feedback on verification automation. When asked about their preference between interactive systems like Veriscope and fully

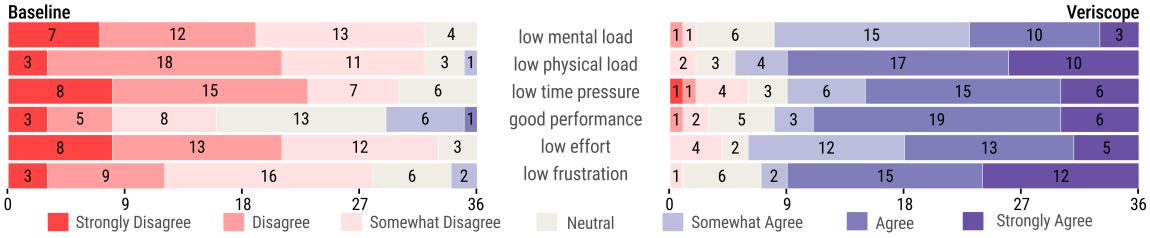


Figure 5: Comparison of the baseline system and Veriscope on subjective ratings.

automated tools driven by LLMs for the verification task, a substantial portion of participants (14/18) favored Veriscope. A number of participants expressed concerns over model hallucinations. P6 stated: "*If the model already makes errors in its analytical insights, how can I trust it to correctly verify its own outputs?*" P18 also emphasized that the results of data analysis could have great impacts on decision-making, mentioning: "*The stake is too high to rely solely on the model.*" Moreover, P17 highlighted that such a human-in-the-loop approach not only improves the accuracy of analytical results, but also enhances users' understanding of data patterns. Although P14 preferred more automated alternatives, she acknowledged that she would probably still use our system to iteratively verify and refine the model's initial results.

8 Technical Evaluation

Veriscope uses LLMs for several features, including artifact summary, insight tagging, and Vega-Lite transformation, with provenance extraction and keyword mapping being the most critical. Other features, like bidirectional highlighting, code tracing, and table queries, rely on deterministic rules or executable artifacts. We conducted an evaluation of the two core LLM-based features' reliability and discussed failure cases and mitigation strategies.

8.1 Evaluation Setup

To systematically evaluate feature reliability across datasets and prompts, we used `gemini-3-flash-preview` to generate 13 analytical results from 5 datasets, with 2–4 user prompts applied to each dataset. These datasets include the Car Sales Report, JHU CSSE COVID-19 Data, and Superstore datasets from the user study, along with two additional datasets: Boston Crimes[7] and Online Shopping[8]. We used a subset of each dataset for this evaluation.

8.2 Empirical Findings

8.2.1 Provenance extraction. We assessed 81 extractions of analytic provenance based on selected claims within generated insights. 80% (65/81) correctly summarized the reasoning steps without omitting critical steps or introducing irrelevant ones. Only 1/81 omitted a critical step. The remaining 15 failures preserved the critical reasoning chain but added 1–2 irrelevant steps. For the majority of summarized steps, Veriscope retrieved at least one critical artifact, such as code, chart, or execution result. In the first summarized step, the model could sometimes fail to cite the first extracted Thought artifact. We employed a post-processing check to address this issue.

8.2.2 Keyword mapping. With Veriscope, each extracted keyword is associated with a specific chart and transformed into values that conform to the corresponding table schema. Across these two stages, keyword-to-chart association reached an accuracy of 99% (293/295) and keyword-value transformation reached an accuracy of 92% (271/293). Transformation failures mainly involved imprecise numeric descriptions such as "over 80%". When the exact statistic is not explicitly represented in the chart schema, Veriscope may return the original numeric text or an estimated range. Meanwhile, if terms like "highest" exist in the sentence, imprecise values could be automatically resolved into max or min values.

8.3 Error Mitigation

Irrelevant steps in the provenance summary do not significantly impact users' understanding of overall reasoning logic, since they introduce no contradictory information and users can discard those parts and their associated artifacts during detailed examination. To further reduce the number of irrelevant steps, our system could scope extraction by the last linked chart position and filter artifacts by data fields. For the less frequent case of missing critical steps, extraction typically captures the step most directly related to a claim while omitting earlier findings it depends on. In this case, users may suspect an omission and our system allows them to unfold preceding reasoning artifacts to recover the earlier analytical process. If relevant artifacts are missing from a summarized step's reference, the same mechanism can be used to unfold them after users navigate to the corresponding context. However, users currently need to proactively inspect the reasoning context to notice missing steps or artifacts. Future work could integrate additional LLMs to flag potential omissions and support users in reviewing and refining the extracted provenance. For imprecise numeric mappings, users can fall back on categorical keywords in the same sentence and verify exact values via Vega-Lite tooltips or table queries.

9 Discussion

Support for varied verification workflows. When validating outputs with Veriscope, users can adopt different strategies according to their individual preferences. One approach is to leverage underlined keywords to perform rapid cross-validation, reserving provenance tracing for cases where particular claims remain uncertain or require closer inspection. Alternatively, users may first extract the analytic provenance to obtain a high-level overview of

[7] https://www.kaggle.com/datasets/ankkur13/boston-crime-data
[8] https://www.kaggle.com/datasets/jacksondivakarr/online-shopping-dataset

the reasoning chain, and then drill down into referenced artifacts based on the generated summary to examine finer-grained details. Regarding the prioritization of artifacts, users can either begin from a specific insight and navigate to its associated artifacts, or leverage the bidirectional highlighting mechanism to start from an intermediate chart and verify whether the key information within the visualization is adequately reflected in the insights. Together, these interaction options highlight the importance of supporting flexible verification workflows that accommodate varied granularity, ranging from high-level summaries to fine-grained artifact details, as well as varying directionality, enabling users to initiate the validation process from either insights or intermediate artifacts.

Interactive verification vs automatic error detection. Our system combines LLM assistance and intuitive interactions to support the verification of LLM-generated results. Through LLM integration, Veriscope substantially improves the efficiency of locating critical information, establishing connections among key artifacts, and extracting coherent reasoning chains. Meanwhile, intuitive interactions and step-by-step summaries of analytic provenance enable users to engage in the verification process with greater ease and fluency. Our human-AI collaborative method preserves verification efficiency while improving accuracy and user trust. In contrast, fully automated error detection, though maximally reducing human effort, introduces considerable risks. For instance, the verification results may misalign with user intent, or inaccurate error localization could mislead users' understanding of the analytical results [19], compromising downstream tasks.

Extension to open-ended verification. The analytical results generated by LLMs contain a mix of correct and incorrect answers. Instead of directly declaring errors, Veriscope leverages various design features to surface verification evidence, such as chart values, code, table queries, and analytic provenance, for users to confirm correct insights and identify incorrect ones. In our user study, we primarily evaluated how effectively the system supported users in identifying errors. Future studies could explore more open-ended verification, where user behaviors include correctly accepting valid outputs as well as making false edits to correct results.

System performance. Before users begin their verification process with Veriscope, our system takes approximately 30 to 50 seconds to process LLM responses and diverse reasoning artifacts, including the keyword extraction and mapping, transforming Python code to Vega-Lite specification, tagging insights with data fields, and summarizing various artifacts. To reduce the time spent on preprocessing, we have adopted three main strategies to optimize our processing pipeline. First, for tasks with no dependencies, we issue parallel requests to the Gemini API. In addition, since summarizing a large number of reasoning artifacts generated from the analytical process can be time-consuming, we partition these artifacts into smaller segments and process them concurrently. Second, we consolidate logically related tasks into a single API call. Third, for simpler tasks, we adjust the model's thinking level to lower processing overhead. These optimizations collectively keep the preprocessing phase within an acceptable time range.

Scalability to longer and multi-turn analytical conversations. In this study, we primarily focus on analytical process conducted in single-turn conversations, where LLMs perform multi-step reasoning. For more complex analytical scenarios, including longer traces with a greater number of reasoning artifacts, and multi-turn analytical conversations [14] in which analysts iteratively refine their analytical goals through follow-up prompts, our existing parallel processing strategy is already designed to scale with larger numbers of artifacts, and remains effective as trace length grows. Cross-artifact connections also allow users to quickly navigate relevant artifacts across long traces. Our system adopts an on-demand approach for the extraction of analytic provenance based on selected claims within insights, avoiding precomputation across all insights. For multi-turn settings, we can bound cost by conditioning on relevant turns rather than the full chat history and scope the session to the last linked artifact, preserving cross-turn continuity while limiting context size. Meanwhile, to improve the effectiveness and user experience of our system in the future, we think it can be useful to support incremental examination of LLM responses through progressive disclosure of reasoning artifacts in the interface [47] and introduce multi-scale representations to enable both overview and detailed exploration of the analytical process. Multi-turn contexts also open up new opportunities for leveraging provenance to steer the reasoning process of LLMs [22, 30].

Accessibility for non-analyst users. While our system is designed to streamline the verification of LLM-generated analytical results, it still presents a non-trivial barrier for users with limited data analysis experience. Such users may lack the contextual familiarity needed to critically assess the analytical outcomes generated by LLMs. This limitation could constrain the accessibility of our system to a broader audience. Therefore, future work could explore how LLMs can be leveraged to lower this barrier by helping non-analyst users develop a basic understanding of dataset features and domain semantics. For instance, LLMs could proactively generate natural language summaries of key dataset properties, or offer contextual explanations tailored to the user's level of familiarity.

10 Conclusion

In this study, we present a claim-centered view of analytic provenance and developed Veriscope, an interactive system that supports provenance-aware verification of LLM-generated analytical results. Based on design considerations derived from a formative study, Veriscope establishes fine-grained connections between insights, charts, code, and data tables, enabling intuitive cross-validation including bidirectional highlighting, code snippet tracing, and keyword-driven table queries. Besides, the system supports on-demand extraction of analytic provenance, presenting the underlying reasoning logic as a navigable, step-by-step summary. Users can further correct identified errors through direct editing or guided regeneration. A user study (N=18) confirmed that Veriscope improves verification efficiency and accuracy while reducing cognitive load compared to a baseline system. The reliability of LLM-based features was assessed through a technical evaluation.

Acknowledgments

We appreciate our reviewers' feedback. This work was supported by New Generation Artificial Intelligence-National Science and Technology Major Project (2025ZD0123503), Zhejiang Provincial Natural Science Foundation of China (LD25F020003), NSFC (62532011), and Ningbo Yongjiang Talent Programme (2024A-399-G).

References

[1] Ian Arawjo, Chelse Swoopes, Priyan Vaithilingam, Martin Wattenberg, and Elena L. Glassman. 2024. ChainForge: A Visual Toolkit for Prompt Engineering and LLM Hypothesis Testing. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. ACM, New York, NY, USA, Article 304, 18 pages. doi:10.1145/3613904.3642016

[2] Forrest Sheng Bao, Miaoran Li, Renyi Qu, Ge Luo, Erana Wan, Yujia Tang, Weisi Fan, Manveer Singh Tamber, Suleman Kazi, Vivek Sourabh, Mike Qi, Ruixuan Tu, Chenyu Xu, Matthew Gonzales, Ofer Mendelevitch, and Amin Ahmad. 2025. FaithBench: A Diverse Hallucination Benchmark for Summarization by Modern LLMs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*. Association for Computational Linguistics, 448–461. doi:10.18653/v1/2025.naacl-short.38

[3] Jan-Micha Bodensohn, Ulf Brackmann, Liane Vogel, Anupam Sanghi, and Carsten Binnig. 2025. Unveiling Challenges for LLMs in Enterprise Data Engineering. *Proc. VLDB Endow.* 19, 2 (2025), 196–209. doi:10.14778/3773749.3773758

[4] Steven P. Callahan, Juliana Freire, Emanuele Santos, Carlos E. Scheidegger, Cláudio T. Silva, and Huy T. Vo. 2006. VisTrails: visualization meets data management. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data* (Chicago, IL, USA) *(SIGMOD '06)*. ACM, New York, NY, USA, 745–747. doi:10.1145/1142473.1142574

[5] Juntong Chen, Jiang Wu, Jiajing Guo, Vikram Mohanty, Xueming Li, Jorge Piazentin Ono, Wenbin He, Liu Ren, and Dongyu Liu. 2025. InterChat: Enhancing Generative Visual Analytics using Multimodal Interactions. *Computer Graphics Forum* 44, 3 (2025), e70112. doi:10.1111/CGF.70112

[6] Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2026. Towards reasoning era: a survey of long chain-of-thought for reasoning large language models. *Science China Information Sciences* 69, 6 (2026), 161101. doi:10.1007/s11432-025-4665-8

[7] Furui Cheng, Vilém Zouhar, Simran Arora, Mrinmaya Sachan, Hendrik Strobelt, and Mennatallah El-Assady. 2024. RELIC: Investigating Large Language Model Responses using Self-Consistency. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. ACM, New York, NY, USA, Article 647, 18 pages. doi:10.1145/3613904.3641904

[8] Victoria Clarke and Virginia Braun. 2017. Thematic analysis. *The Journal of Positive Psychology* 12, 3 (2017), 297–298. doi:10.1080/17439760.2016.1262613

[9] A.M. Dean. 2001. Experimental Design: Overview. In *International Encyclopedia of the Social & Behavioral Sciences*. Pergamon, Oxford, 5090–5096. doi:10.1016/B0-08-043076-7/00417-4

[10] Ensheng Dong, Hongru Du, and Lauren Gardner. 2020. An interactive web-based dashboard to track COVID-19 in real time. *The Lancet Infectious Diseases* 20, 5 (2020), 533–534. doi:10.1016/S1473-3099(20)30120-1

[11] Klaus Eckelt, Kiran Gadhave, Alexander Lex, and Marc Streit. 2025. Loops: Leveraging Provenance and Visualization to Support Exploratory Data Analysis in Notebooks. *IEEE Transactions on Visualization and Computer Graphics* 31, 1 (2025), 1213–1223. doi:10.1109/TVCG.2024.3456186

[12] K. Gadhave, Z. Cutler, and A. Lex. 2024. Persist: Persistent and Reusable Interactions in Computational Notebooks. *Computer Graphics Forum* 43, 3 (2024), e15092. doi:10.1111/CGF.15092

[13] Katy Ilonka Gero, Chelse Swoopes, Ziwei Gu, Jonathan K. Kummerfeld, and Elena L. Glassman. 2024. Supporting Sensemaking of Large Language Model Outputs at Scale. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. ACM, New York, NY, USA, Article 838, 21 pages. doi:10.1145/3613904.3642139

[14] Ken Gu, Srishti Palani, and Vidya Setlur. 2026. "I Need to Find That One Chart": How Data Workers Navigate, Summarize and Communicate Analytical Conversations. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)*. ACM, New York, NY, USA, Article 1145, 33 pages. doi:10.1145/3772318.3790802

[15] Ken Gu, Ruoxi Shang, Tim Althoff, Chenglong Wang, and Steven M. Drucker. 2024. How Do Analysts Understand and Verify AI-Assisted Data Analyses?. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. ACM, New York, NY, USA, Article 748, 22 pages. doi:10.1145/3613904.3642497

[16] Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Human Mental Workload*. Advances in Psychology, Vol. 52. North-Holland, 139–183. doi:10.1016/S0166-4115(08)62386-9

[17] Jiayi Hong, Christian Seto, Arlen Fan, and Ross Maciejewski. 2025. Do LLMs Have Visualization Literacy? An Evaluation on Modified Visualizations to Test Generalization in Data Interpretation. *IEEE Transactions on Visualization and Computer Graphics* 31, 10 (2025), 7004–7018. doi:10.1109/TVCG.2025.3536358

[18] Matt-Heun Hong and Anamaria Crisan. 2025. Data Has Entered the Chat: How Data Workers Conduct Exploratory Visual Analytic Conversations with GenAI Agents. *ACM Trans. Interact. Intell. Syst.* 15, 4, Article 21 (2025), 40 pages. doi:10.1145/3744750

[19] Jiaming Ji, Tianyi Qiu, Boyuan Chen, Jiayi Zhou, Borong Zhang, Donghai Hong, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Lukas Vierling, Zhaowei Zhang, Fanzhi Zeng, Juntao Dai, Xuehai Pan, Hua Xu, Aidan O'Gara, Kwan Ng, Brian Tse, Jie Fu, Stephen Mcaleer, Yanfeng Wang, Mingchuan Yang, Yunhuai Liu, Yizhou Wang, Song-Chun Zhu, Yike Guo, Yaodong Yang, and Wen Gao. 2025. AI Alignment: A Contemporary Survey. *ACM Comput. Surv.* 58, 5, Article 132 (2025), 38 pages. doi:10.1145/3770749

[20] Peiling Jiang, Jude Rayan, Steven P. Dow, and Haijun Xia. 2023. Graphologue: Exploring Large Language Model Responses with Interactive Diagrams. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23)*. ACM, New York, NY, USA, Article 3, 20 pages. doi:10.1145/3586183.3606737

[21] Minsuk Kahng, Ian Tenney, Mahima Pushkarna, Michael Xieyang Liu, James Wexler, Emily Reif, Krystal Kallarackal, Minsuk Chang, Michael Terry, and Lucas Dixon. 2025. LLM Comparator: Interactive Analysis of Side-by-Side Evaluation of Large Language Models. *IEEE Transactions on Visualization and Computer Graphics* 31, 1 (2025), 503–513. doi:10.1109/TVCG.2024.3456354

[22] Majeed Kazemitabaar, Jack Williams, Ian Drosos, Tovi Grossman, Austin Zachary Henley, Carina Negreanu, and Advait Sarkar. 2024. Improving Steering and Verification in AI-Assisted Data Analysis with Interactive Task Decomposition. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology* (Pittsburgh, PA, USA) *(UIST '24)*. ACM, New York, NY, USA, Article 92, 19 pages. doi:10.1145/3654777.3676345

[23] Florian Leiser, Sven Eckhardt, Valentin Leuthe, Merlin Knaeble, Alexander Mädche, Gerhard Schwabe, and Ali Sunyaev. 2024. HILL: A Hallucination Identifier for Large Language Models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. ACM, New York, NY, USA, Article 482, 13 pages. doi:10.1145/3613904.3642428

[24] Yunxin Li, Zhenyu Liu, Zitao Li, Xuanyu Zhang, Zhenran Xu, Xinyu Chen, Haoyuan Shi, Shenyuan Jiang, Xintong Wang, Jifang Wang, Shouzheng Huang, Xinping Zhao, Borui Jiang, Lanqing Hong, Longyue Wang, Zhuotao Tian, Baoxing Huai, Wenhan Luo, Weihua Luo, Zheng Zhang, Baotian Hu, and Min Zhang. 2025. Perception, Reason, Think, and Plan: A Survey on Large Multimodal Reasoning Models. arXiv:2505.04921 [cs.CV] https://arxiv.org/abs/2505.04921

[25] Karthic Madanagopal, Eric D. Ragan, and Perakath C. Benjamin. 2019. Analytic Provenance in Practice: The Role of Provenance in Real-World Visualization and Data Analysis Environments. *IEEE Computer Graphics and Applications* 39, 6 (2019), 30–45. doi:10.1109/MCG.2019.2933419

[26] Arpit Narechania, Adam Coscia, Emily Wall, and Alex Endert. 2022. Lumos: Increasing Awareness of Analytic Behavior during Visual Data Analysis. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2022), 1009–1018. doi:10.1109/TVCG.2021.3114827

[27] Arpit Narechania, Shunan Guo, Eunyee Koh, Alex Endert, and Jane Hoffswell. 2025. Utilizing Provenance as an Attribute for Visual Data Analysis: A Design Probe With ProvenanceLens. *IEEE Transactions on Visualization and Computer Graphics* 31, 10 (2025), 8452–8465. doi:10.1109/TVCG.2025.3571708

[28] Arpit Narechania, Kaustubh Odak, Mennatallah El-Assady, and Alex Endert. 2025. ProvenanceWidgets: A Library of UI Control Elements to Track and Dynamically Overlay Analytic Provenance. *IEEE Transactions on Visualization and Computer Graphics* 31, 1 (2025), 1235–1245. doi:10.1109/TVCG.2024.3456144

[29] Chaoxu Pang, Yixuan Cao, Chunhao Yang, and Ping Luo. 2024. Uncovering Limitations of Large Language Models in Information Seeking from Tables. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics, 1388–1409. doi:10.18653/v1/2024.findings-acl.82

[30] Rock Yuren Pang, K. J. Kevin Feng, Shangbin Feng, Chu Li, Weijia Shi, Yulia Tsvetkov, Jeffrey Heer, and Katharina Reinecke. 2026. Interactive Reasoning: Visualizing and Controlling Chain-of-Thought Reasoning in Large Language Models. In *Proceedings of the 31st International Conference on Intelligent User Interfaces (IUI '26)*. ACM, New York, NY, USA, 852–867. doi:10.1145/3742413.3789091

[31] Uwe Peters and Benjamin Chin-Yee. 2025. Generalization bias in large language model summarization of scientific research. *Royal Society Open Science* 12, 4 (2025), 241776. doi:10.1098/rsos.241776

[32] Eric D. Ragan, Alex Endert, Jibonananda Sanyal, and Jian Chen. 2016. Characterizing Provenance in Visualization and Data Analysis: An Organizational Framework of Provenance Types and Purposes. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2016), 31–40. doi:10.1109/TVCG.2015.2467551

[33] Ali Sarvghad, Melanie Tory, and Narges Mahyar. 2017. Visualizing Dimension Coverage to Support Exploratory Analysis. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 21–30. doi:10.1109/TVCG.2016.2598466

[34] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2017. Vega-Lite: A Grammar of Interactive Graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 341–350. doi:10.1109/TVCG.2016.2599030

[35] Holger Stitz, Samuel Gratzl, Harald Piringer, Thomas Zichner, and Marc Streit. 2019. KnowledgePearls: Provenance-Based Visualization Retrieval. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 120–130.

doi:10.1109/TVCG.2018.2865024

[36] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23)*. ACM, New York, NY, USA, Article 1, 18 pages. doi:10.1145/3586183.3606756

[37] Yuan Tian, Weiwei Cui, Dazhen Deng, Xinjing Yi, Yurun Yang, Haidong Zhang, and Yingcai Wu. 2025. ChartGPT: Leveraging LLMs to Generate Charts From Abstract Natural Language. *IEEE Transactions on Visualization and Computer Graphics* 31, 3 (2025), 1731–1745. doi:10.1109/TVCG.2024.3368621

[38] Yuan Tian, Chuhan Zhang, Xiaotong Wang, Sitong Pan, Weiwei Cui, Haidong Zhang, Dazhen Deng, and Yingcai Wu. 2025. ReSpark: Leveraging Previous Data Reports as References to Generate New Reports with LLMs. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology (UIST '25)*. ACM, New York, NY, USA, Article 181, 18 pages. doi:10.1145/3746059.3747644

[39] Chenglong Wang, Bongshin Lee, Steven M. Drucker, Dan Marshall, and Jianfeng Gao. 2025. Data Formulator 2: Iterative Creation of Data Visualizations, with AI Transforming Data Along the Way. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. ACM, New York, NY, USA, Article 677, 17 pages. doi:10.1145/3706598.3713296

[40] Chenglong Wang, John Thompson, and Bongshin Lee. 2024. Data Formulator: AI-Powered Concept-Driven Visualization Authoring. *IEEE Transactions on Visualization and Computer Graphics* 30, 1 (2024), 1128–1138. doi:10.1109/TVCG.2023.3326585

[41] Xingqi Wang, Yiming Cui, Xin Yao, Shijin Wang, Guoping Hu, and Xiaoyu Qin. 2025. ChartHal: A Fine-grained Framework Evaluating Hallucination of Large Vision Language Models in Chart Understanding. arXiv:2509.17481 [cs.CV] https://arxiv.org/abs/2509.17481

[42] Yaoting Wang, Shengqiong Wu, Yuecheng Zhang, Shuicheng Yan, Ziwei Liu, Jiebo Luo, and Hao Fei. 2025. Multimodal Chain-of-Thought Reasoning: A Comprehensive Survey. arXiv:2503.12605 [cs.CV] https://arxiv.org/abs/2503.12605

[43] Zhijie Wang, Zijie Zhou, Da Song, Yuheng Huang, Shengmai Chen, Lei Ma, and Tianyi Zhang. 2025. Towards Understanding the Characteristics of Code Generation Errors Made by Large Language Models. In *2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE)*. IEEE, 2587–2599. doi:10.1109/ICSE55347.2025.00180

[44] Luoxuan Weng, Xingbo Wang, Junyu Lu, Yingchaojie Feng, Yihan Liu, Haozhe Feng, Danqing Huang, and Wei Chen. 2025. InsightLens: Augmenting LLM-Powered Data Analysis With Interactive Insight Management and Navigation. *IEEE Transactions on Visualization and Computer Graphics* 31, 6 (2025), 3719–3732. doi:10.1109/TVCG.2025.3567131

[45] Yifan Wu, Joseph M. Hellerstein, and Arvind Satyanarayan. 2020. B2: Bridging Code and Interactive Visualization in Computational Notebooks. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. ACM, New York, NY, USA, 152–165. doi:10.1145/3379337.3415851

[46] 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*. Association for Computational Linguistics, 12174–12200. doi:10.18653/v1/2024.findings-emnlp.710

[47] Liwenhan Xie, Chengbo Zheng, Haijun Xia, Huamin Qu, and Chen Zhu-Tian. 2024. WaitGPT: Monitoring and Steering Conversational LLM Agent in Data Analysis with On-the-Fly Code Visualization. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology* (Pittsburgh, PA, USA) *(UIST '24)*. ACM, New York, NY, USA, Article 119, 14 pages. doi:10.1145/3654777.3676374

[48] Zijun Yao, Yantao Liu, Yanxu Chen, Jianhui Chen, Junfeng Fang, Lei Hou, Juanzi Li, and Tat-Seng Chua. 2025. Are Reasoning Models More Prone to Hallucination? arXiv:2505.23646 [cs.CL] https://arxiv.org/abs/2505.23646

[49] Wenshuo Zhang, Leixian Shen, Shuchang Xu, Jindu Wang, Jian Zhao, Huamin Qu, and Lin-Ping Yuan. 2025. NeuroSync: Intent-Aware Code-Based Problem Solving via Direct LLM Understanding Modification. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology (UIST '25)*. ACM, New York, NY, USA, Article 30, 19 pages. doi:10.1145/3746059.3747668

[50] Ziyao Zhang, Chong Wang, Yanlin Wang, Ensheng Shi, Yuchi Ma, Wanjun Zhong, Jiachi Chen, Mingzhi Mao, and Zibin Zheng. 2025. LLM Hallucinations in Practical Code Generation: Phenomena, Mechanism, and Mitigation. *Proc. ACM Softw. Eng.* 2, ISSTA, Article ISSTA022 (2025), 23 pages. doi:10.1145/3728894

[51] Yuheng Zhao, Xueli Shu, Liwen Fan, Lin Gao, Yu Zhang, and Siming Chen. 2026. ProactiveVA: Proactive Visual Analytics with LLM-Based UI Agent. *IEEE Transactions on Visualization and Computer Graphics* 32, 1 (2026), 451–461. doi:10.1109/TVCG.2025.3642628

[52] Yuheng Zhao, Junjie Wang, Linbing Xiang, Xiaowen Zhang, Zifei Guo, Cagatay Turkay, Yu Zhang, and Siming Chen. 2025. LightVA: Lightweight Visual Analytics With LLM Agent-Based Task Planning and Execution. *IEEE Transactions on Visualization and Computer Graphics* 31, 9 (2025), 6162–6177. doi:10.1109/TVCG.2024.3496112

[53] Yuheng Zhao, Yixing Zhang, Yu Zhang, Xinyi Zhao, Junjie Wang, Zekai Shao, Cagatay Turkay, and Siming Chen. 2025. LEVA: Using Large Language Models to Enhance Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics* 31, 3 (2025), 1830–1847. doi:10.1109/TVCG.2024.3368060

[54] Jiajun Zhu, Xinyu Cheng, Zhongsu Luo, Yunfan Zhou, Xinhuan Shu, Di Weng, and Yingcai Wu. 2025. ViseGPT: Towards Better Alignment of LLM-generated Data Wrangling Scripts and User Prompts. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology (UIST '25)*. ACM, New York, NY, USA, Article 147, 16 pages. doi:10.1145/3746059.3747689