Linking Event-sequence Data Claims to Analytic Provenance in Chatbots: Technique and Evaluation

Po-Ming Law
Adaptive Machines, Inc
San Francisco, California, USA
terrance@adaptivemachines.ai

Jianheng Ouyang*
Nguyen-Truong Thinh*
The Hong Kong University of Science and Technology
Hong Kong, China
jouyangae@connect.ust.hk
truongthinh.nguyen03@gmail.com

Weizhi Li
Adaptive Machines, Inc
San Francisco, California, USA
weizhi@adaptivemachines.ai

Arpit Narechania
The Hong Kong University of Science and Technology
Hong Kong, China
arpit@ust.hk

Abstract

Chatbots are increasingly used to analyze temporal event sequences of which website clickstreams are among the most common. Explanations are essential for understanding claims about event-sequence data. However, existing systems often require programming expertise to interpret these explanations and fail to explain the data processes underlying the data claims. We introduce a technique for linking a data claim to its analytic provenance (i.e., the analytic steps and results that produced it). It has two components: a reference table that records analytic steps and results in addressable cells, and a citation format that binds each claim to one or more cells. To operationalize this technique for clickstream analysis, we curated a set of analytic operations for clickstream analysis, organized them through a pivot-table abstraction, and developed tools that an AI agent can invoke to create a reference table and assemble a natural language response with cited data claims. To evaluate how citations affect users' assessment of data claims, we conducted two crowdsourced experiments with 90 participants each. The experiments showed that participants perceived correct data claims to be more accurate with citations when compared with no explanations, and that they perceived incorrect data claims to be less accurate with citations when compared with code and no explanations. **Supplementary materials:** https://prototypes-production-efcc.up.railway.app

CCS Concepts

• **Human-centered computing** → **User interface design**; **Natural language interfaces**.

1 Introduction

Event-sequence data are extensively collected across diverse domains, from healthcare [44] to marketing [40]. Clickstreams are among the most widely collected forms of behavioral data [36]. Any site or application instrumented with a web analytics platform produces them continuously. While many consumer chatbots and AI agents can automatically analyze this data, understanding the natural language responses could be challenging.

Consider a product manager who uploads some clickstream data to a consumer chatbot such as Gemini [11] and asks, *"Are there any noticeable friction points among our international mobile shoppers?"* (Figure 1). The dataset consists of a set of events. Each event records a session id, a timestamp, and the page visited. These events can be grouped by the session id and ordered by the timestamp to reconstruct a session (each visitor's sequence of interactions). Each session carries attributes such as device type and country.

The chatbot writes Python, runs it, and reports that South Africa stands out: *"...The main drop-off in the complete conversion funnel occurs between placing an item in the cart and initiating checkout, where 31.30% of users abandon their sessions."*

The sentence invites two questions. Which events make up the *"complete conversion funnel"*? And 31.30% of whom—of all South African mobile visitors, or only of the narrower cohort who placed an item in their cart? The product manager clicks "Show code" (Figure 1 left) to inspect the generated code. In Figure 1 right, the top code snippet computes the percentage of South African mobile users who added items to their shopping cart (67.3% users did that). The bottom snippet then computes the percentage of users who reached the checkout stage (36% users) and the percentage of users who abandoned their carts (67.3% - 36% = 31.3%). Reassembling a claim from computations spread across snippets is laborious even for programmers, and this product manager does not program at all.

There are three considerations when designing an explanation technique that answers the product manager's questions. First, the explanations should not require users to read code. Existing systems that decompose AI-generated analysis into editable plans [19] or interactive visualizations [46] often target users proficient in programming.

Second, the explanations should provide the analytic provenance (i.e., analytic steps and results) behind the data claims (e.g., *"31.30%"*) within a natural language response. While some techniques offer stepwise explanations for AI-generated visualizations [45], they do not support linking text-based data claims to computational steps.

Third, the explanation technique should support event-sequence data. Existing tools like Aletheia [10] provide citations for tabular data claims but cannot generalize to event sequences. These tools are often built on low-level analytic task taxonomies for tabular data

*Both authors contributed equally to this research.

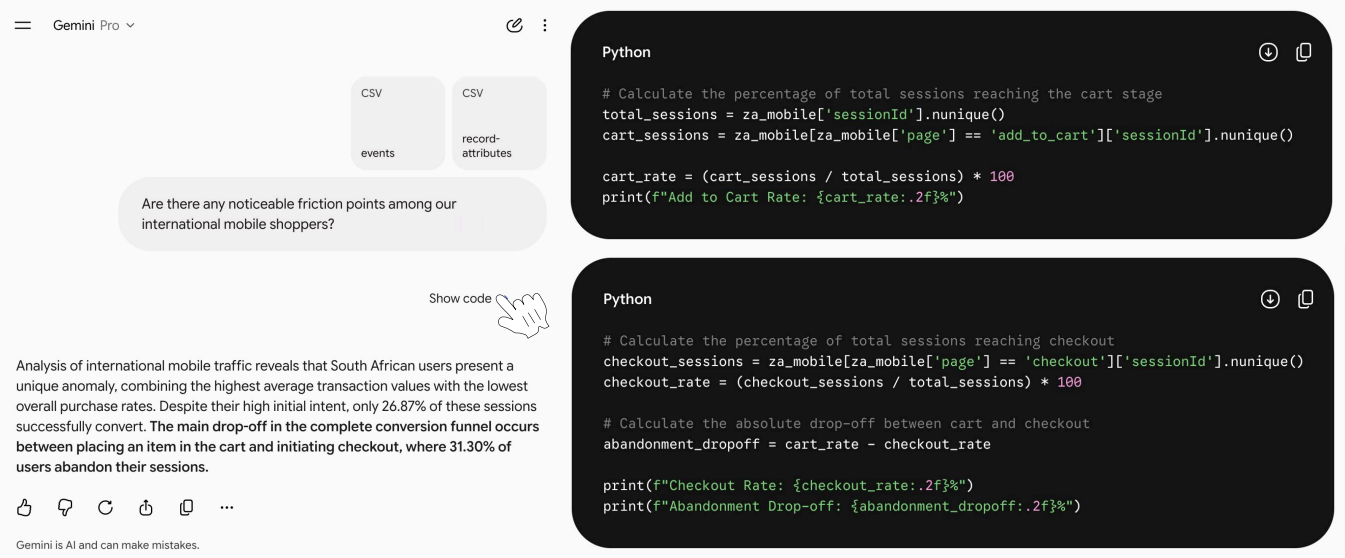


Figure 1: The prompt and response between a user and the Gemini application. This example comes from an actual chat session but has been simplified for clarity. The user asks, *"Are there any noticeable friction points among our international mobile shoppers?"* The chatbot writes Python code and returns a natural-language response. Being curious about the analytic steps, the user clicks on "Show code" to reveal the generated Python code on the right.

(e.g., [3]) while event sequences involve temporal patterns, funnels, and sequential behaviors and have their own task frameworks [8, 14, 52].

To address these limitations, we ask, *"How can we link an event-sequence data claim in a natural language response to the analytic steps and results that produced it, without requiring users to read code?"* We introduce a technique for linking event-sequence data claims to their analytic provenance. The technique has two components, one for representing provenance and one for linking claims to it. For representing provenance, we introduce a reference table. Each cell in this reference table records a sequence of analytic steps together with the results the steps produced. For linking claims to provenance, we introduce a citation format. Using this format, the chatbot wraps each data claim in its response in a tag referencing the cell elements that claim rests on.

Figure 2 shows the effect. The product manager hovers over *"31.30%"* and a floating panel displays the analytic steps and results. In the explanation, the system first filters the sessions to focus on those using a mobile device from South Africa. It then evaluates a funnel with a page view event, an add to cart event, a checkout, and finally a purchase. This funnel clarifies that *"complete conversion funnel"* refers to this sequence of four events. The funnel highlights that 67.3% of all South African mobile users added items to their carts and 36% of South African mobile users checked out. 31.3% is the difference between 67.3% and 36%, indicating that 31.3% of all South African mobile users abandoned their carts without checking out. A single hover addresses both of the product manager's questions without reading code.

To operationalize the technique, we curated a set of analytic operations for clickstream analysis based on event-sequence literature [8, 14, 52] and expert consultations. We organized them through a pivot-table abstraction [41] and developed a set of analytical functions that an AI agent can invoke to create a reference table with analytic provenance and link data claims in its response to the provenance. Following the terminology used in the AI agent community, we call these functions "tools" [38, 48]. These tools support operations such as selecting sessions by device or country, filtering sessions based on ordered event patterns, segmenting sequences between events, mining frequent patterns, and performing funnel analysis.

We demonstrate the technique in two applications: automated insight generation (Figure 6a) and post-hoc fact-checking (Figure 6b). In the prototype for automated insight generation, users ask analytic questions and the chatbot answers with cited data claims. Many chatbots offer to fact-check a natural language response (Figure 3), but they target facts checkable against the web. In our prototype for post-hoc fact-checking, users enter event-sequence data claims. The system then verifies the claims against the data and adds inline citations.

An explanation, however, is not automatically an improvement. Explanations can cause users to over-rely on AI output even when the output is wrong [20]. Hence, we ask, *"How do citations of analytic provenance shape users' assessment of data claims?"* To address this question, we conducted two between-subject crowdsourced experiments corresponding to the two applications. In each experiment, 90 participants reviewed a series of data claims with no explanations, code snippets, or citations.

Experiment 1 examined automated insight generation in a setting where all presented claims were correct. We found that when a chatbot produced perfectly accurate insights, participants tended to perceive correct data claims with citations to be more accurate compared to no explanations. Experiment 2 examined post-hoc



Figure 2: Using our technique, a user can hover over a data claim to see the analytic steps and results. Here, hovering over *"31.30%"* reveals that the system filters to sessions on mobile devices and from South Africa. The system then conducts a funnel analysis and finds that 67.3% of all South African mobile users added items to their carts and 36% checked out. 31.3% is the difference between 67.3% and 36%.

fact-checking in a setting where some claims contained controlled errors. When a chatbot occasionally produced errors, participants appeared to perceive incorrect data claims with citations to be less accurate compared to both code and no explanations.

Together, the experiments reveal an asymmetry: increasing perceived accuracy of correct claims while reducing perceived accuracy of incorrect ones. A central goal of explaining event-sequence data claims is to promote appropriate reliance: users trust AI-generated analyses when they are correct and reject them when they are not [26]. This asymmetry is consistent with the pattern that appropriate reliance requires.

Our primary contributions include:

(1) A technique for linking event-sequence data claims to their analytic provenance, comprising a reference-table representation of analytic steps and results and a citation format that binds claims within a natural-language response to cells of that table.
(2) An operationalization of the technique for clickstream analysis: we curated nine analytic operations, organized them through a pivot-table abstraction and developed tools that AI agents can call to create a reference table of analytic provenance.
(3) Two prototypes demonstrating the technique for automated insight generation and for post-hoc fact-checking.
(4) Findings from two crowdsourced experiments that compare how citations, code, and no explanations affect users' assessments of data claims within a chatbot's responses.

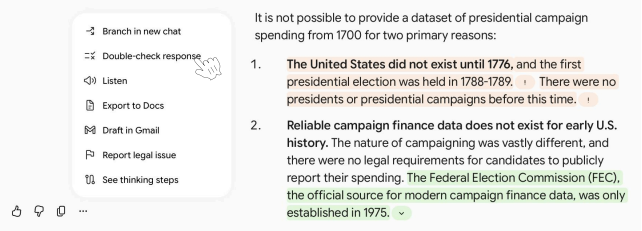


Figure 3: In the Gemini web application, users can select *"Double-check response"* for certain responses. The system will highlight dubious claims in red and accurate claims in green. Users can click on a highlighted claim to check the sources.

2 Background and Related Work

This paper draws on common terminology in the engineering of LLM-based agent systems, including agent and tool. Because these terms are used inconsistently across research and practice, we first define the terminology in this paper.

Our technique uses a set of tools that an agent can invoke to perform analytic operations on event sequence data. Hence, we review the operations commonly employed by systems for analyzing temporal event sequences. We further build on work that explains and fact-checks AI-generated analyses, and on approaches for evaluating such explanations.

2.1 Terminology

2.1.1 LLM. The AI agent engineering community often treats an LLM as a model endpoint: a stateless service that maps an input prompt to a text completion.

2.1.2 Agent. An agent consists of a "harness" (i.e., infrastructure) built around the LLM. The harness includes system instructions that govern the LLM's behaviors and a set of "tools" (i.e., functions) that the LLM can invoke. With the harness, the agent calls tools to obtain results, feeds the results back to the LLM, which calls the tools in a control loop until the LLM produces a final answer [48].

2.1.3 Tool. Tools are functions that an agent can invoke. Some tools call external services or APIs. For instance, a web-search tool can send a query to a search engine and return the results. When a tool integrates with such external services, the interface is often standardized through the Model Context Protocol (MCP). Other

tools perform purely local computation and require no external API calls. For example, a tool can be a simple function that takes two numbers and returns their sum.

Tools also differ in whether they have side effects. For example, the tool that calculates the sum of two numbers does not have side effects. Tools that alter some external state when invoked have side effects. Consider a coding agent that operates within a sandbox: it may call a tool to write a new file to the sandbox and commit the changes to a GitHub repository.

2.1.4 Chatbot. A chatbot consists of a user interface built around an agent.

2.1.5 Data Claim. We use data statement to refer to a sentence generated by an LLM based on the results of a data analysis. A data claim is a factual span within that statement whose meaning is supported by one or more analysis results. A data claim may report an exact value. For example, in *"67.3% of sessions involve adding an item to the cart,"* the claim *"67.3%"* is derived from a funnel analysis. A data claim may also express a qualitative characterization of magnitude or frequency. For example, in *"A substantial proportion of sessions involve adding an item to the cart,"* the claim *"a substantial proportion"* is grounded in the 67.3% obtained from the same funnel analysis.

Our technique provides an agent with tools for performing analytic operations on temporal event sequences. We group these operations into three pivot-table operations [41]: row operations, column operations, and filter operations. These tools do not make external API calls but have side effects: the agent operates within a workspace containing a reference table, and each tool invocation can add to or update that table. A chatbot built around this agent can associate event-sequence data claims in a natural-language response with entries in the reference table. When a user interacts with a cited claim, the interface displays the corresponding analytical steps and results (as in Figure 2).

2.2 Analytic Operations in Event Sequence Analysis

Researchers have developed visual analytic systems for analyzing event sequences in different application domains. For instance, Coreflow [30] enables marketers to explore clickstreams and understand customer journeys. EventFlow [33] supports investigating electronic health records while MAQUI [25] facilitates the exploration of emergency department workflow. These systems have been designed for different analytic purposes including cohort comparison [22, 32], pattern mining [31, 42], and outcome prediction [13, 23].

Several efforts have surveyed these systems and identified the analytic operations involved in event-sequence analysis. Notably, Du et al. [8] describe 15 strategies for reducing data volume and pattern variety during event-sequence analysis. Guo et al. [14] categorized event-sequence analysis systems by five high-level analytical tasks. Zinat et al. [52] developed a hierarchical framework for describing event sequence analysis tasks. Across these accounts, recurring operations include filtering event sequences (e.g., extracting clickstream sessions from South Africa), segmenting sequences (e.g., focusing only on the events that happen after checkout), and mining frequent patterns.

Our technique exposes a set of tools that an agent can invoke to perform analytic operations on event sequences. These operations are grounded in the existing literature and feedback from an expert event sequence analyst.

2.3 Explaining AI-Generated Data Claims

Explanations of AI-generated answers take various forms depending on the application. They may aim to improve transparency [28], support comprehension [5, 18], or help users verify correctness [19, 46]. When answering factual inquiries (e.g., who is the president of the United States), AI systems may provide supporting details [27, 47] or citations to external sources [21, 29] alongside the answer.

For data claims, Kazemitabaar et al. [19] explored breaking an analysis into editable assumptions, plan, and code. Xie et al. [46], on the other hand, improved the readability of the generated code using interactive visualizations. Both explanation approaches target users proficient in programming. They also do not map a data claim within a natural language response to the corresponding analytic steps. StepMIND provides multimodal, stepwise explanations for AI-generated visualizations [45]. However, it does not link individual claims in a textual response to their analytic provenance and does not address event-sequence analysis.

Besides research systems, consumer chatbots (e.g., [4, 11, 34]) often present Python code to explain AI-generated data claims. As we discussed in the introduction, this approach presents two challenges: 1) it is difficult to trace which computations in the code support a data claim, and 2) understanding the code could be out of reach for users who lack programming expertise. We propose a technique for citing event-sequence data claims in a chatbot's interface and displaying the analytic provenance when users hover over a claim. Our proposed explanations do not require reading code.

2.4 Fact-Checking AI-Generated Data Claims

Automated fact checking has traditionally focused on verifying factual claims (e.g., Donald Trump is the 47th US president) that are not produced through data analysis (e.g., analyzing a CSV file). The veracity of factual claims can be automatically checked against knowledge bases such as Wikipedia pages [39] and scientific articles [43]. For data claims, verification relies on inferences from data. Previous efforts translate claims into SQL queries to validate them against relational databases [17] or process chart images to check data statements visually [2].

Consumer chatbots such as the Gemini application enable users to double-check factual claims (see Figure 3). Dubious facts are then highlighted in red and correct facts in green. Users can click on a highlighted fact to check the sources. Inspired by this use case, we developed an application in which users enter event-sequence data claims and push a button to fact-check the claims (Figure 6b). The system cites the claims and users can hover over a claim to see the analytic steps. Closest to our work is Aletheia, which uses an LLM pipeline to parse data claims into structured fact specifications and executes those specifications against a reference dataset [10].

Aletheia targets tabular data, however, and does not support verifying event-sequence data claims.

2.5 Evaluating Explanations for AI-Generated Data Claims

Prior research has evaluated various types of explanations for AI-generated analyses. Examples include communicating accuracy information [16, 49], explanations of outputs [6, 37], onboarding materials [7, 24], and debugging interventions [15]. Such evaluations often ask whether an explanation fosters appropriate reliance: broadly, users trust AI-generated analyses when they are correct and reject them when they are not [26].

Our crowdsourced experiments found that the citations improved perceived accuracy of correct event-sequence data claims compared to no explanations and reduced perceived accuracy of incorrect claims compared to both code and no explanations. Most relevant to our work, Kim et al. [20] found that clickable sources in LLM responses increase appropriate reliance on correct answers and reduce overreliance on incorrect ones, especially when paired with inconsistency highlighting. Their study examined citations to webpages for general factual questions. We instead studied the effect of showing analytic steps and results on users' assessment of data claims during event sequence analysis.

3 Linking Event-Sequence Data Claims to Analytic Provenance

Consider the statement from the introduction: *"the main drop-off in the complete conversion funnel occurs between placing an item in the cart and initiating checkout, where 31.3% of users abandon their sessions"*. Calculating the abandonment rate of 31.3% involves 1) filtering the sessions by mobile devices, 2) filtering to those from South Africa, 3) performing a funnel analysis, and 4) computing the drop-off from adding to cart to checkout (see the four steps in Figure 2).

Our goal is to retain these analytic steps and results and associate them with the corresponding claim in the chatbot response. When users hover over *"31.3%,"* the chatbot can then reveal which sessions were analyzed, how the funnel was defined, and how the value was calculated.

In doing so, we designed a technique to store the analytic steps and results in a reference table. In this section, we will discuss how an AI agent can invoke tools to construct this reference table (Sec. 3.1). We will then showcase two use cases: automatic insight generation (Sec. 3.2) and post-hoc fact-checking (Sec. 3.3).

3.1 Constructing Reference Table

Based on the survey of event sequence analytic operations [8, 14, 52] and feedback from an expert event sequence analyst, we curated a set of analytic operations for analyzing clickstream data. We organized these analytic operations through a pivot-table abstraction [41] and developed tools that an AI agent invokes to perform operations on event sequences. Using these tools, the agent constructs a reference table in which each cell records the analytic steps and results.

3.1.1 Analytic Operations. Zinat et al. [52] identify a wide range of event-sequence analytic operations, highlighting that the relevance of these operations varies across application domains. Hence, as opposed to creating a system that supports all possible analytic operations, we focused on the operations in a specific application domain while ensuring that our framework is extensible: to support new applications, we should be able to easily plug in new analytic operations into our framework. We will discuss the extensibility of our technique in Section 6.1.

To identify analytic operations in a specific application domain, we collaborated closely with an expert analyst who has more than ten years of experience in event sequence analysis. At the time of our collaboration, the analyst worked at an e-commerce platform company and regularly analyzed clickstream data. We met weekly over two months. Earlier meetings focused on understanding the analyst's data and workflow. Subsequent meetings focused on refining the analytic operations in our framework. During these meetings, we presented a set of operations, obtained feedback, and refined the set until no additional changes were proposed.

Because the collaborator's data are proprietary, we cannot discuss their data in this paper. Their data have a similar structure to a Kaggle e-commerce clickstream dataset [9]. We will use this dataset as a running example.

In this Kaggle dataset, an event represents a user interaction and can be defined as a set of attribute-value pairs: $E = \{A_1 = V_1, A_2 = V_2, \ldots, A_j = V_j\}$. It has a session id, a timestamp, and attributes such as page (with values including `page_view`, `add_to_cart`, and `purchase`). We call the attributes associated with an event event attributes.

An event sequence (or a session) is an ordered list of events: $S = [E_1, E_2, \ldots, E_k]$. It is constructed by grouping events with the same session id and ordering them by timestamp. Each event sequence has attributes describing the properties of the sequence. We call the attributes associated with a sequence record attributes. Example record attributes include device (e.g., `mobile`, `desktop`, and `tablet`) and source (e.g., `paid`, `organic`, and `direct`).

Our collaborator described a recurring two-stage data analysis workflow: extracting a cohort (e.g., extracting the sessions for South African mobile users) and performing computations on the cohort (e.g., performing funnel analysis on South African mobile sessions). In the following, we discuss the detailed analytic operations employed by our collaborator in this workflow. We do not claim that these operations are exhaustive, but they reflect the analytic requirements of our collaborator. We also note that our framework is extensible: new operations for other applications can be implemented as functions and plugged into the framework easily.

Here are the operations for defining cohorts:

Filter by numerical record attributes. This operation selects sequences whose numerical record attribute falls within a specified range. For example, it can select sessions associated with users between 25 and 40 years old.

Filter by categorical record attributes. This operation selects sequences whose categorical attribute matches a specified value. For example, it can select sessions where device = `mobile`.

Filter by patterns. It selects sequences that contain a specified pattern. An example pattern is `page_view` → `add_to_cart`. Events in this pattern do not need to be contiguous. Filtering with this pattern returns the sessions in which a `page_view` event occurs before an `add_to_cart` event, regardless of intervening events.

Patterns may include negative conditions. An example is `page_view` → `add_to_cart` → `NOT purchase`. Filtering with this pattern returns the sessions in which a `page_view` event occurs before an `add_to_cart` event. However, there is no `purchase` event after the `add_to_cart` event. These sessions correspond to shoppers who added items to their cart but never completed a purchase.

Segment by patterns. It extracts a segment within each event sequence. For the pattern `add_to_cart` → `checkout`, this operation first selects sequences where `checkout` occurs after `add_to_cart`. For these sequences, it truncates each to include only the events between the matching `add_to_cart` and `checkout` events. An analyst can then investigate, for example, the common pattern between adding an item to the cart and initiating checkout.

Here are the operations for performing computations on the cohorts:

Aggregate by numerical record attributes. With a set of event sequences, we can then compute aggregate statistics including average, sum, min, and max over a numerical record attribute. For instance, each session is associated with the age of the shopper. We can find the minimum age across sessions.

Aggregate by categorical record attributes. With a set of event sequences, we can find the distribution of a categorical record attribute. For example, each session is associated with a device type. This operation returns the percentage of sessions associated with mobile, desktop, and tablet devices.

Mine frequent patterns. It mines the top-k most frequent sequential patterns along with the percentage of sequences containing each pattern. An example result is that the pattern `page_view` → `add_to_cart` → `checkout` occurs in 35% of the selected sessions. Our framework employs the PrefixSpan algorithm [35], which is commonly used for sequential pattern mining.

Analyze event attribute distribution. This operation calculates the distribution of an event attribute across all events in the selected sequences. For example, it can report the percentages of events corresponding to `add_to_cart` and `checkout`. It may return that page = `checkout` appears in 1% of the events, indicating that 1% of the page views are the checkout pages.

Analyze funnel. It involves defining a funnel (i.e., a sequence of events) for the analysis. For each defined step, this operation calculates: (1) the conversion rate (the percentage of sequences reaching the step), (2) the average duration between steps, and (3) the average number of intervening events between steps. For example, in Figure 2, 67.3% of all sessions have an `add_to_cart` event. The average duration between `add_to_cart` and checkout is 51 minutes and 46 seconds.

3.1.2 Pivot-Table Abstraction. The pivot-table abstraction aligns well with our goals to G1) enable an AI agent to perform analysis on event sequences, G2) assemble a natural language response with data claims, and G3) link each data claim to the concrete analytic steps that are not code.

There are three types of operations in a pivot table: Row operations add rows that represent entities for the analysis; column operations perform computations on each entity; filter operations apply additional filtering constraints to each cell in the pivot table.

A conventional pivot table illustrates the distinction. Suppose a sales dataset contains one record per transaction, including the salesperson and sales amount. A row operation can create one row per salesperson while a column operation can create a column for each salesperson's total sales. In Figure 4, the first cell indicates that Mira has total sales of $100.

	SUM(sales)
Salesperson = Mira	$100
Salesperson = Peter	$80

Figure 4: A simple pivot table. A row operation creates one row per salesperson and a column operation creates a column for each salesperson's total sales.

From this example, the pivot table supports data exploration without writing code (aligns with G1). Each cell in the table represents a finding that can be described in natural language (e.g., Mira's total sales is $100) (aligns with G2). A cell also represents concrete analytic steps for a finding (aligns with G3). For instance, in Figure 4, the first cell is obtained by filtering the records for Mira and computing the total sales for these records.

Besides aligning with our goals, the pivot table abstraction corresponds well with the two types of analytic operations we identified. The four operations for defining cohorts correspond to row operations in the pivot table while the five operations for performing computations correspond to column operations. Filter operations comprise the event sequence analytic operations involving filtering sequences or events, which are the same set as the four operations for defining cohorts. This results in four row operations, five column operations, and four filter operations in our framework.

Figure 5 shows how we can use these operations to create a reference table. We first add a row that represents organic-traffic sessions (Figure 5a STEP 1). These sessions are extracted by filtering using a categorical record attribute (source = `organic`). At this point, the reference table has rows but no populated cells because no column has been added. Next, we add another row that represents subsequences occurring between `page_view` and `add_to_cart` (Figure 5a STEP 2). These sessions are extracted by segmenting using the `page_view` and `add_to_cart` pattern.

We then apply a column operation to find the average age of both cohorts (Figure 5a STEP 3). This operation results in a JSON object representing the analytic steps and results for the organic traffic sessions (Figure 5b). The JSON object shows that analytic step 1 filters sequences where source = `organic`, yielding 40,776 sessions. The type of `categorical-record-attribute- filter` indicates that this step involves filtering by a categorical record attribute. Analytic step 2 calculates the average age, yielding 47. The type of `numerical- record-attribute-aggregation` indicates that this step involves aggregating by a numerical record attribute.

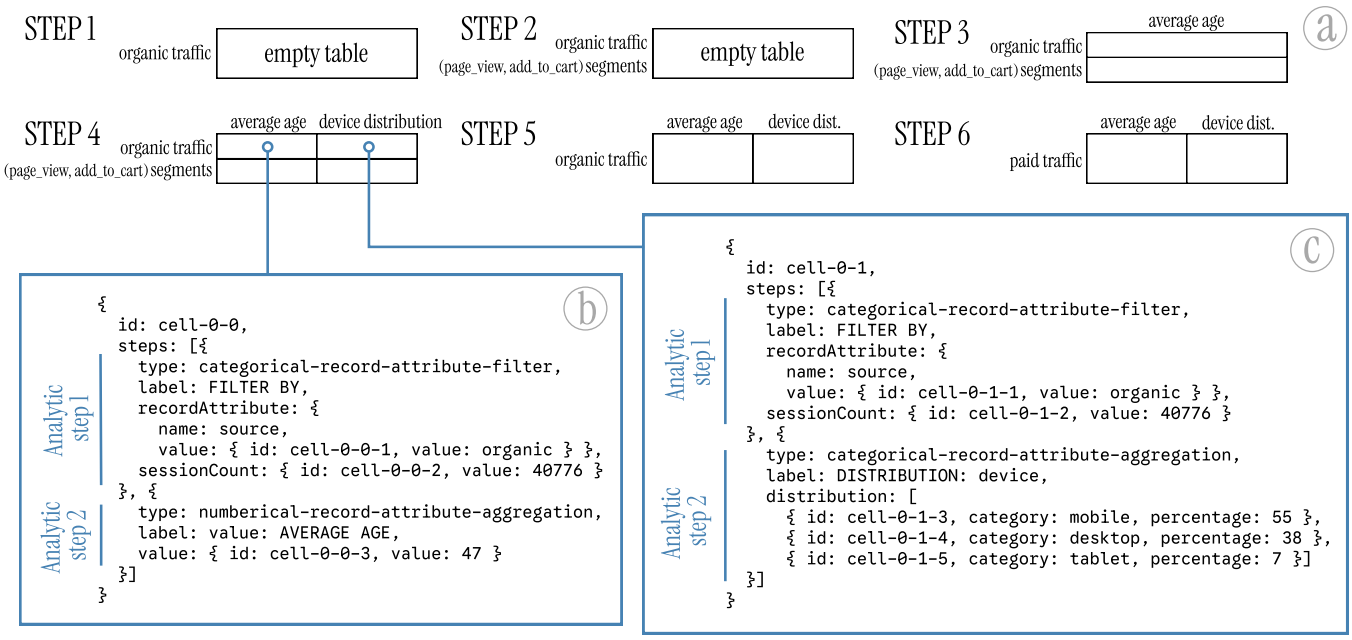


Figure 5: Reference table construction. In (a), step 1 adds a row for organic-traffic sessions; step 2 adds a row for segments between `page_view` and `add_to_cart`; step 3 adds a column to calculate the average age for both cohorts; step 4 adds a device distribution column; step 5 removes the second row; step 6 changes the first row to paid-traffic sessions. The top-left cell in step 4 represents the average age of users associated with organic-traffic sessions (b). In (b), analytic step 1 filters by source = `organic`, yielding 40,776 sessions, while analytic step 2 calculates the average age, yielding 47. The top-right cell in step 4 represents the device distribution for organic traffic (c), where analytic step 1 is filtering and step 2 calculates the percentages of devices.

We then apply another column operation to find the device distribution (Figure 5a STEP 4). This operation results in another JSON object in Figure 5c. The JSON object shows that analytic step 1 filters sequences where source = `organic` while analytic step 2 calculates that 55% of sessions used mobile, 38% used desktop, and 7% used tablet.

3.1.3 Tools for Performing Analytic Operations. Section 3.1.1 describes the event sequence analytic operations within our framework. Section 3.1.2 illustrates how we organize the analytic operations into row, column, and filter operations to construct a reference table. AI agents interact with the external environment by calling tools. How do we package the row, column, and filter operations into tools?

We provide the agent with a sandbox environment. In this sandbox, the agent can mutate the reference table by calling three tools: the add tool applies row, column, or filter operations to add rows and columns to the reference table or to filter it; the delete tool removes any applied row, column, or filter operation; the update tool modifies an applied operation.

In Figure 5a, an agent constructs a reference table in these steps:

Step 1: Invoke the add tool to add a row representing organic-traffic sessions.
Step 2: Invoke the add tool to add another row representing subsequences between `page_view` and `add_to_cart`.
Step 3: Invoke the add tool to add a column calculating the average age for both rows.
Step 4: Invoke the add tool to add another column calculating the device distribution for both rows.
Step 5: Invoke the delete tool to remove the row for subsequences between `page_view` and `add_to_cart`.
Step 6: Invoke the update tool to change the remaining row from source = `organic` to source = `paid`.

3.2 Automatic Insight Generation

The above illustrates how an AI agent constructs a reference table. In this reference table, a cell has some analytic steps and the associated data analysis results. From the data analysis results, the agent can assemble a natural language response with data claims. It can link each claim to a cell. When users hover over a claim linked to a cell, the chatbot displays the analytic provenance in that cell.

The first use case of this technique is automatic insight generation. We developed a chatbot prototype (Figure 6a). Users can ask analytic questions about the underlying event-sequence data and the chatbot responds with cited data claims. Users can hover over a cited claim to visualize the analytic steps that generated the claim.

Below, we discuss how the agent links a claim to the reference table and how the chatbot visualizes the referenced provenance.

3.2.1 Citations. Elements (the cell itself and values) in a cell in the reference table have ids. In the system prompt, we instruct

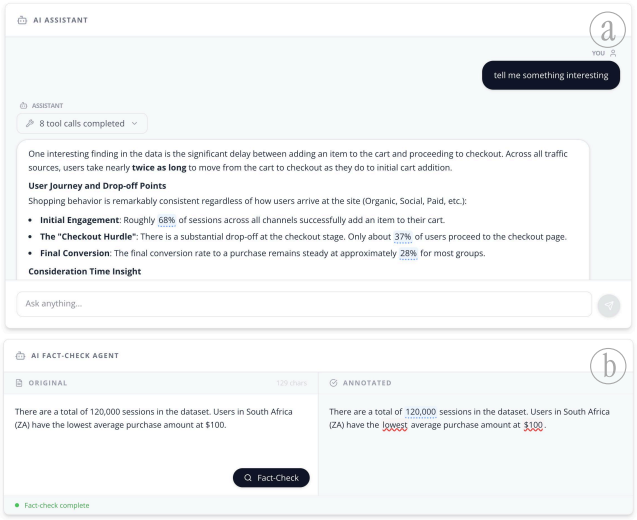


Figure 6: Automated insight generation (a) and post-hoc fact-checking (b). Users can hover over a cited claim and see the analytic steps and results in a floating panel.

the agent to wrap each data claim in an XML `<cite>` tag that includes a `referenceIds` attribute. The AI agent references the id of cell elements to link a data claim to the corresponding analytic provenance.

As an example, in Figure 7, suppose the user asks, *"Are there any noticeable friction points among our international mobile shoppers?"* and the agent constructs the reference table in Figure 7a. The agent then responds "`<cite referenceIds="cell-0-0-6">67.3%</cite>` *of South African mobile users added items to their cart."* (Figure 7b left). The id `cell-0-0-6` corresponds to the value of 67.3% sessions that added items to the cart. By referencing `cell-0-0-6`, the agent links the claim to the reference table cell that contains the analytic provenance.

The `referenceIds` attribute can contain multiple ids linked by arithmetic or logical operators (+, −, ×, ÷, >, <, =, ≈, and ,). In Figure 7b right, the agent further responds that, *"The main drop-off in the complete conversion funnel occurs between placing an item in the cart and initiating checkout, where* `<cite referenceIds="cell-0-0-6 - cell-0-0-7">31.30%</cite>` *of users abandon their sessions."* `cell-0-0-6` corresponds to the percentage of users who added items to their cart while `cell-0-0-7` corresponds to the percentage of users who checked out. `cell-0-0-6 - cell-0-0-7` represents the difference between the two values.

3.2.2 Visualizations. Within a natural language response, the chatbot highlights cited claims in blue. When users hover over a citation, a floating panel displays the corresponding analytical provenance.

If the agent cites an entire cell, the panel displays all analytical steps and results stored in that cell. If the agent cites a particular element within a cell, the panel displays the complete reference while visually emphasizing that element. For example, in Figure 7b left, the claim cites the value associated with the `add_to_cart` step in the funnel. The floating panel highlights the `add_to_cart` step.

If a claim cites multiple ids, the panel visualizes each element in order and displays the resulting calculation. For example, in Figure 7b right, the panel highlights the `add_to_cart` event and the `checkout` event in the funnel. It further displays the difference between the two percentages at the end.

Figure 8 provides example visualizations of the nine analytic operations in Section 3.1.1 as steps in the floating panel:

Filter by numerical record attributes. As an example, selecting sessions associated with users between 25 and 40 years old is visualized as FILTER BY AGE 25 - 40 (Figure 8a step 1).

Filter by categorical record attributes. Selecting sessions from South Africa is visualized as FILTER BY SESSIONCOUNTRY ZA (Figure 8c step 1).

Filter by patterns. Selecting sessions that have a `page_view` event and no `purchase` events afterward is visualized as FILTER BY PAGE page_view → PAGE ✕ purchase (Figure 8b step 1).

Segment by patterns. Segmenting by `add_to_cart` → `checkout` is visualized as first filtering sessions with the pattern `add_to_cart` → `checkout` FILTER BY PAGE add_to_cart → PAGE checkout (Figure 8b step 1) and second extracting the events between `add_to_cart` and `checkout` FOCUS BETWEEN PAGE add_to_cart AND PAGE checkout (Figure 8b step 2).

Aggregate by numerical record attributes. The panel displays an aggregated value directly. Figure 8c step 2 shows the minimum age across all selected sessions.

Aggregate by categorical record attributes. The panel shows a bar graph with the percentages for different record attribute categories. Figure 8a step 2 shows the percentage of devices across sessions.

Mine frequent patterns. The panel shows the patterns together with the percentages of sequences with the pattern. Figure 8e step 3 shows the top-5 frequent patterns. 100% of sessions have the single-event pattern `page_view`.

Analyze event attribute distribution. The panel displays a bar graph showing the percentages for different event attribute categories. Figure 8d step 1 shows that 70.9% of events are `page_view` events.

Analyze funnel. The panel displays a funnel showing the percentage reaching each step and the duration or the number of events between steps. Figure 8b step 2 shows a two-step funnel with an `add_to_cart` event and then a `checkout` event.

3.3 Post-Hoc Fact-Checking

The second use case of our technique is post-hoc fact-checking. Consumer chatbots such as the Gemini application allow users to double-check factual statements after a response has been generated (Figure 3). Similarly, once an agent generates event-sequence data claims, we can run the claims through a second agent to fact-check the claims using our technique.

To demonstrate this use case, we developed an automated fact-checking prototype (Figure 6b). Users enter event-sequence data claims and click on a button to invoke fact-checking. After the verification, the system highlights correct claims in blue and incorrect

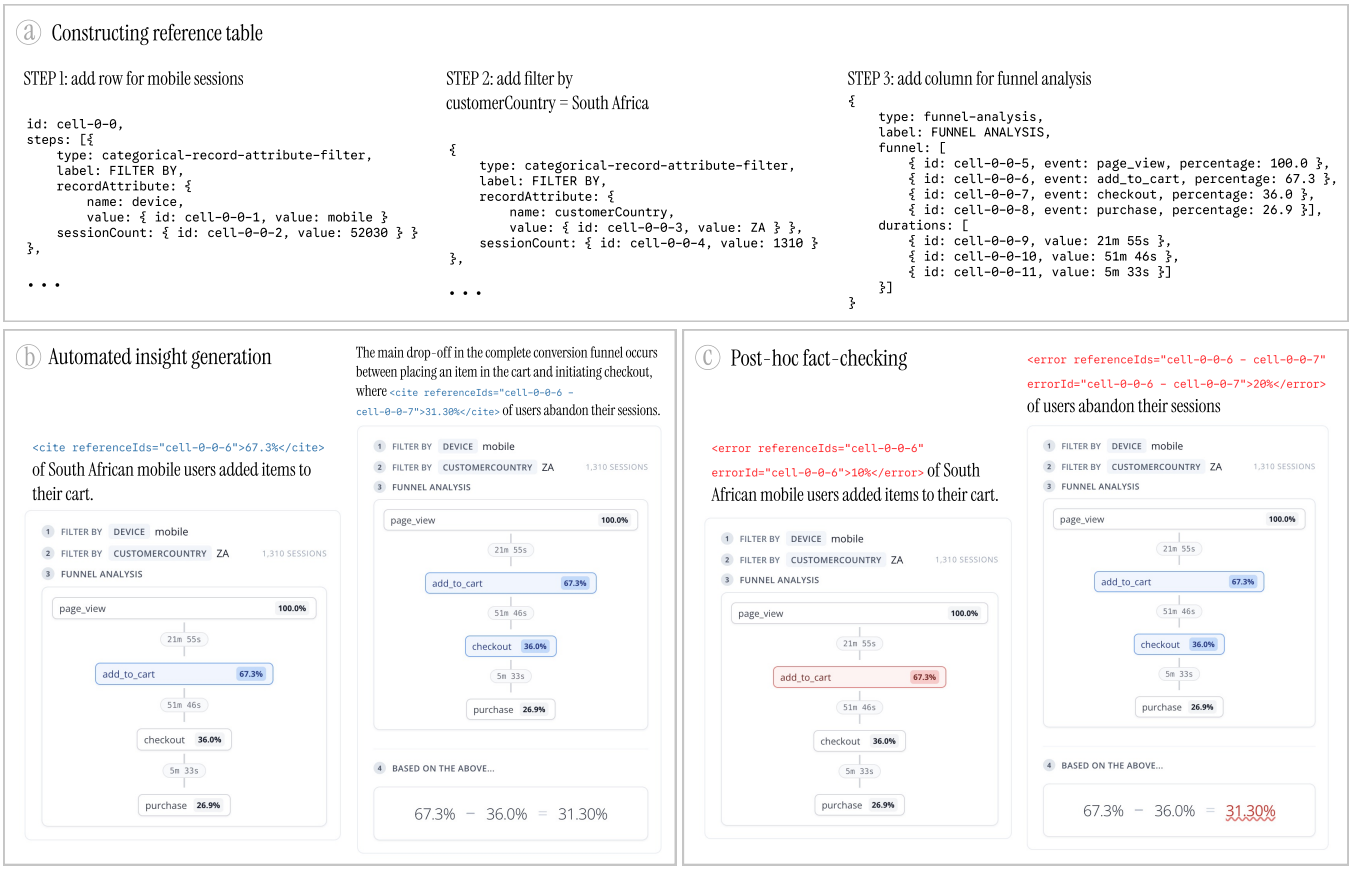


Figure 7: Using a reference table for automated insight generation and post-hoc fact-checking. In (a), a reference table is constructed by adding a row for mobile sessions (step 1), selecting only South African sessions (step 2), and adding a column for funnel analysis (step 3). There is only one cell in the table. The JSON objects in steps 1-3 combined show the complete data in the cell. In (b) left, a user hovers over *"67.3%"* in *"67.3% of South African mobile users added items to their cart"* and a funnel with the `add_to_cart` step highlighted is shown. In (b) right, a user hovers over *"31.3%"* in *"31.30% of users abandon their sessions."* *"31.3%"* is the percent difference between sessions reaching the `add_to_cart` step and the `checkout` step. These two steps are highlighted and a calculation of the difference is displayed. In (c) left, the `add_to_cart` step is highlighted in red to show the discrepancy between the *"10%"* in the claims and the calculated percentage of *"67.3%"* sessions reaching the `add_to_cart` step. In (c) right, the difference of *"31.3%"* is highlighted in red to indicate that the cart abandonment percentage is not the claimed 20%.

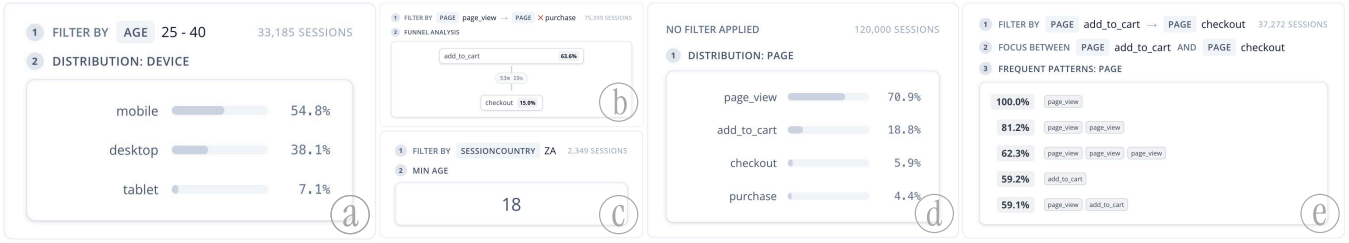


Figure 8: Example visualizations of the nine analytic operations in the floating panel.

ones in red. Users can hover over either type of claim to inspect the underlying analytic steps and results.

3.3.1 Citations. The system prompt instructs the AI agent to wrap each correct data claim with a `<cite>` tag. It wraps each incorrect data claim with an `<error>` tag containing a referenceIds attribute and an errorId attribute. The `referenceIds` attribute identifies the

references relevant to evaluating the claim. The `errorId` attribute identifies the value or relationship that conflicts with the stated claim.

For the statement *"10% of South African mobile users added items to their cart,"* the agent constructs the table in Figure 7a and embeds an `<error>` tag around 10%: *"`<error referenceIds="cell-0-0-6" errorId="cell-0-0-6">10%</error>` of South African mobile users added items to their cart."* The `errorId` points to the calculated value of 67.3%, making the discrepancy with the stated 10% explicit.

For the statement *"20% of users abandon their sessions,"* the agent embeds an `<error>` tag around 20%: *"`<error referenceIds="cell-0-0-6 - cell-0-0-7" errorId="cell-0-0-6 - cell-0-0-7">20%</error>` of users abandon their sessions."* The `errorId` indicates a discrepancy between 20% and the calculated 67.3% - 36% = 31.3%.

3.3.2 Visualizations. The interface highlights correct claims in blue and incorrect claims in red. Hovering over a claim opens the same floating panel used for automatic insight generation.

In the floating panel, the element referenced by `errorId` is highlighted in red to make the discrepancy between the claim and the data salient. For *"`<error referenceIds="cell-0-0-6" errorId="cell-0-0-6">10%</error>` of South African mobile users added items to their cart,"* `cell-0-0-6` corresponds to the `add_to_cart` step in the funnel and we highlight this step in red (Figure 7c left). For *"`<error referenceIds="cell-0-0-6 - cell-0-0-7" errorId="cell-0-0-6 - cell-0-0-7">20%</error>` of users abandon their sessions,"* `cell-0-0-6 - cell-0-0-7` corresponds to the difference, and we highlight the difference in red (Figure 7c right).

4 Crowdsourced Experiment 1

One use case of our technique is automated insight generation: given some data, an agent analyzes the data, synthesizes a natural language response, and cites the event-sequence data claims in the response. The rapid advancement of LLMs suggests a future where the reliability of AI-generated analyses is significantly higher. This raises a critical question: if all data claims are perfectly accurate, how do the citations shape users' assessments of the claims?

To investigate this, we designed a between-subjects experiment in which participants reviewed a series of correct event-sequence data claims under one of three explanation conditions: citations, code snippets, or no explanations (baseline). We measured perceived accuracy of insights, perceived overall system reliability, and system understanding.

4.1 Methodology

4.1.1 Fact Preparation. We used the same Kaggle e-commerce dataset described in Section 3.1.1 [9]. We chose this dataset because its domain (i.e., online shopping behavior) is accessible to laypeople without specialized training.

To prepare the citation condition, we loaded this dataset into our chatbot prototype (Figure 6a) and prompted it using the query "I want to get a sense for the data. Tell me some interesting facts about the data." We aimed to simulate a typical chatbot interaction where a user hovers over one citation, views the analytic steps, and forms an assessment of the claim. Hence, we designed each task to present a single statement with one citation and one explanation panel. From the natural language response from our chatbot prototype, we collected ten statements each with exactly one cited data claim.

With the 10 statements, we prepared the code condition next. We supplied Claude Code with the Kaggle dataset and instructed it to write Python code snippets that would produce the corresponding data claims. During the prompting, we followed the common practice where one snippet can generate multiple data claims. As a result, Claude Code produced four code snippets that collectively generated the 10 claims. All snippets include inline comments to improve comprehension. Since chatbots like the Gemini application display console output for each code snippet, we executed each snippet and collected the console output. In the end, we obtained four code snippets and four console outputs for the 10 data claims.

4.1.2 Conditions. We compared three explanation types: no explanations, code snippets and citations.

No explanations (Figure 9a). Participants reviewed ten statements one by one with no supporting information. This condition served as the baseline. For each task, we asked participants to rate the perceived accuracy of a statement. In the task interface, we presented all statements. The statement being rated was highlighted while the other nine were grayed out.

Code snippets (Figure 9b). Chatbots like the Gemini application provide code snippets together with the natural language response even when users may not be proficient in programming. In this condition, participants reviewed the same ten statements with four Python code snippets displayed alongside. In the task interface, we presented all statements on the left with the currently rated one being highlighted. We presented all four code snippets and the respective console outputs on the right.

Citations (Figure 9c). Participants reviewed the same ten statements with one cited data claim each. In the task interface, a cursor pointed to the cited data claim in the statement being rated. We displayed the corresponding analytic steps and results in a floating panel on the right.

4.1.3 Participants. A power analysis indicated that detecting a medium effect size ($f = 0.25$) with a one-way ANOVA at $\alpha = 0.05$ and power of 0.8 required 90 participants (30 per condition).

We recruited participants from Prolific with the following inclusion criteria: an approval rating of 95% or above, English proficiency, a work function related to data analysis, and prior experience using AI chatbots. We did not require programming experience because we targeted the user base of consumer chatbots such as Gemini and ChatGPT, which generate data claims with code snippets regardless of users' coding ability. To screen for attentiveness, we asked 4 comprehension questions about event sequence analysis before the main task; participants who answered fewer than 2 correctly were excluded. They were compensated £13.59 per hour on average.

Participants (61 male, 29 female) were aged 18–24 (15), 25–34 (43), 35–44 (15), 45–54 (13), and 55–64 (4). For data analysis expertise, 24 reported beginner-level, 34 intermediate, 29 advanced, and 3 expert experience. For chatbot usage, 42 used chatbots daily, 31 often, 16 sometimes, and 1 rarely. Among participants in the code condition, 3 had no coding experience, 14 beginner, 8 intermediate, and 5 advanced.

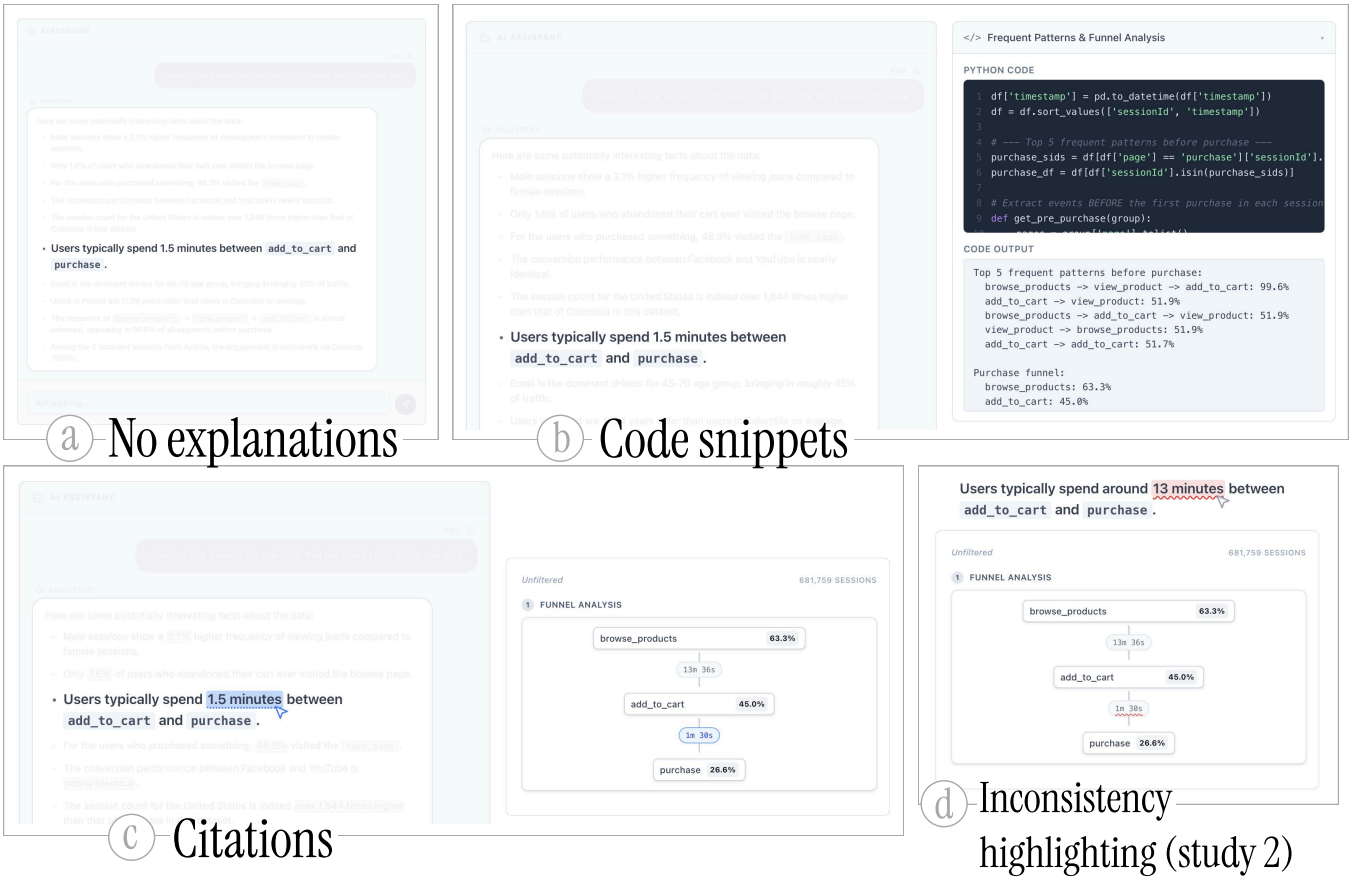


Figure 9: Experimental conditions for experiments 1 and 2. In the no explanation condition (a), the interface for each task shows all 10 statements. Only the statement being rated is highlighted while the rest are grayed out. In the code condition (b), participants saw a statement highlighted on the left and code snippets presented on the right. In the citation condition (c), participants saw a statement on the left and a floating panel with explanatory analytic steps and results on the right. In Experiment 2, for incorrect statements, the data claim is highlighted in red on the left and the floating panel highlights the inconsistency on the right (d).

4.1.4 Procedure. We randomly assigned participants to one of the three conditions. Participants first completed a demographic survey and the screening questions. They then completed a practice task to familiarize themselves with the interface. The practice task mirrored the main task but used different statements.

In the main task, participants reviewed ten statements with data claims. They rated one statement at a time. The presentation order of the 10 statements was randomized to avoid order effects. For each statement, we asked, *"Based on the above information, how accurate or inaccurate does the above statement seem to you?"* Participants reported their perceived accuracy on a 7-point Likert scale.

After the main task, participants answered two post-task questions. First, they rated the overall system reliability on a 7-point Likert scale and provided reasons for their rating. Second, they rated their understanding about the system on a 7-point scale and detailed their justifications in an open-ended question. The study took approximately 23 minutes on average.

4.1.5 Measures. We collected three measures:

Perceived accuracy. For each statement, participants rated *"Based on the above information, how accurate or inaccurate does the above statement seem to you?"* on a 7-point Likert scale (−3: very inaccurate to +3: very accurate). For each participant, we computed the average rating.

Perceived overall reliability. After participants rated the ten statements, we asked, *"Overall, how reliable do you think the facts generated by this chatbot are?"* on a 7-point Likert scale (−3: very unreliable to +3: very reliable).

System understanding. Besides perceived overall reliability, we further asked, *"How well do you understand the steps the AI agent took when generating the insights?"* on a 7-point scale (−3: did not understand to +3: understand completely).

For each quantitative measure, we conducted Kruskal-Wallis tests to examine the main effect of explanation type. When a significant main effect was found, we conducted post-hoc Wilcoxon

rank-sum tests with Holm-Bonferroni correction for pairwise comparisons.

For the two post-task questions regarding perceived overall reliability and system understanding, we also asked participants to explain their ratings in open-ended questions. We conducted a thematic analysis [1] of these qualitative responses to complement the quantitative findings.

4.1.6 Hypotheses. Schemmer et al. [37] showed that explanations tend to increase reliance on AI advice. While their study focused on AI advice and not data analysis, we posited that the finding would carry over to event-sequence data claims.

H1.1. Participants with explanations (citations or code) will report higher perceived accuracy than those without explanations.

H1.2. Participants with explanations (citations or code) will report higher perceived overall reliability than those without explanations.

Xie et al. [46] found that structured visual representations of analytic steps are more comprehensible than raw code. Participants in the citation condition reviewed analytic steps visualized in a floating panel. We expected them to be able to understand the explanations of data claims better than participants in the code condition and the baseline.

H1.3. Participants with citations will report better system understanding than those with code or no explanations.

4.2 Results

Figure 10 shows the quantitative results for the two crowdsourced experiments. We observed that explanations (code or citations) tended to increase perceived accuracy and perceived reliability compared to no explanations. Also, citations led to significantly better system understanding than code and no explanations.

In the following, we provide the detailed analysis.

4.2.1 Perceived Accuracy (H1.1 Supported). Participants in the citation condition rated the statements the most accurate (M=2.55, SD=0.53), followed by the code condition (M=2.44, SD=0.54), and the no-explanation baseline (M=0.93, SD=1.16). A Kruskal-Wallis test indicated a significant main effect of explanation type on perceived accuracy ($\chi^2(2) = 40.80$, p<.001). Post-hoc pairwise comparisons showed that both the citation condition (p<.001) and the code condition (p<.001) had significantly higher perceived accuracy than the baseline. The citation and code conditions did not differ significantly (p=.33).

4.2.2 Perceived Overall Reliability (H1.2 Supported). Participants in the citation condition reported the highest perceived reliability (M=2.60, SD=0.62), followed by the code condition (M=2.40, SD=0.56), and the no-explanation condition (M=1.10, SD=1.45). Explanation type had a significant main effect ($\chi^2(2) = 27.27$, p<.001). Both the citation condition (p<.001) and the code condition (p<.001) had significantly higher perceived reliability than the no-explanation condition. The citation and code conditions did not differ significantly (p=.12).

4.2.3 System Understanding (H1.3 Supported). Participants in the citation condition reported the highest system understanding (M=2.40, SD=0.86), followed by the code condition (M=1.47, SD=1.74), and the no-explanation condition (M=1.00, SD=1.72). Explanation type had a significant main effect ($\chi^2(2) = 13.99$, p<.001). The citation condition had significantly higher system understanding than both the code condition (p=.033) and the no-explanation condition (p<.001). The code and no-explanation conditions did not differ significantly (p=.22).

5 Crowdsourced Experiment 2

Another use case of our technique is post-hoc fact-checking. During post-hoc fact-checking, an agent first writes code to analyze the data and synthesize a natural language response. Similar to the workflow to double-check a claim in Figure 3, users then trigger fact-checking, and the agent leverages our technique to verify and cite the data claims within the response. For correct data claims, users see the supporting data highlighted in blue in the floating panel (as in Figure 7b left) while for incorrect data claims, users see the inconsistency highlighted in red in the panel (as in Figure 7c left).

Experiment 1 shows that for correct data claims, citations tend to improve perceived accuracy when compared with no explanations. For incorrect data claims, how do the citations with inconsistency highlighting affect users' assessments of the claims?

To investigate this, we conducted a second between-subjects experiment. As in the first experiment, participants reviewed a series of statements under the three explanation conditions. However, four of the ten statements contained errors.

5.1 Methodology

5.1.1 Fact Preparation. We started with the same ten statements from Experiment 1. Based on research on common failure modes of AI-generated data analyses, we introduced errors in four of them. Common errors include misinterpretation of computed values [51] and omission of analytic steps such as filters [46].

For misinterpretation errors, the code produces a correct result but the AI agent generates an incorrect data claim. For instance, the code correctly computes that the session count for the United States is 1,844 times higher than that of Colombia. However, the agent rounds 1,844 to 2,000 and wrongly states that "The session count for the United States is indeed over 2,000 times higher than that of Colombia in this dataset."

For missing-filter errors, the code misses a filtering step, results in an incorrect calculation, and hence, an incorrect data claim. For example, an agent attempts to compute the source distribution for the 45–70 age group. It omits the age filter and computes the source distribution across all ages instead. It claims that *"45% of traffic for the 45–70 age group is from email"* when the correct statement is *"45% of traffic for all age groups is from email."*

For the citation condition, we did not change the six correct statements. For the two incorrect statements with interpretation errors, we updated the statement text. For example, we changed *"the session count for the United States is indeed over 1,844 times higher than that of Colombia in this dataset"* to *"the session count for the United States is indeed over 2,000 times higher than that of Colombia in this dataset."* For the two incorrect statements with missing filters, we removed a filter from the floating panel. For

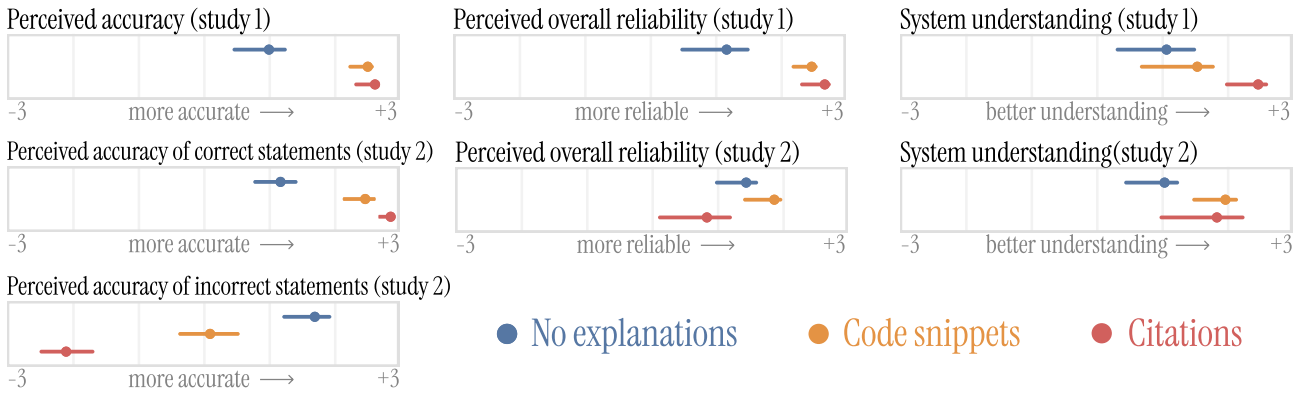


Figure 10: Quantitative results from crowdsourced experiments 1 and 2. Error bars show 95% bootstrapped confidence intervals.

the statement *"email is the dominant driver for the 45–70 age group, bringing in roughly 45% of traffic,"* we removed the step that selects sessions from users aged 45–70. We highlight both the data claim in the incorrect statement and the inconsistency in the floating panel in red (Figure 9d).

For the code condition, we again did not change the six correct statements. For the two incorrect statements with interpretation errors, we updated the statement text without changing the code logic. For the two incorrect statements with missing filters, we removed a filtering step in the code.

5.1.2 Conditions, Participants, and Procedure. The three conditions and the experimental procedure were identical to Experiment 1. We recruited 90 new participants from Prolific (30 per condition) using the same inclusion criteria. Participants were aged 18–24 (10), 25–34 (47), 35–44 (21), 45–54 (7), and 55–64 (5). 48 identified as male and 42 as female. Regarding data analysis expertise, 26 reported beginner-level experience, 33 intermediate, 23 advanced, and 8 expert. For chatbot usage, 52 used chatbots daily, 27 often, 6 sometimes, and 5 rarely. Among participants in the code condition, 5 had no coding experience, 11 beginner, 11 intermediate, 1 advanced, and 2 expert. Participants were compensated £13.02 per hour on average. The study took approximately 24 minutes on average.

5.1.3 Measures. We collected four measures. Perceived overall reliability and system understanding were the same as in Experiment 1. For perceived accuracy, we calculated two measures:

Perceived accuracy of incorrect statements. For each statement, participants rated *"Based on the above information, how accurate or inaccurate does the above statement seem to you?"* on a 7-point scale. For each participant, we computed the average rating over the four incorrect statements.

Perceived accuracy of correct statements. For each participant, we computed the average perceived accuracy rating over the six correct statements.

We measured the perceived accuracy of incorrect statements since we were interested in whether citations with inconsistency highlighting would reduce the perceived accuracy of erroneous statements. We measured the perceived accuracy of correct statements since we wanted to confirm the findings in Experiment 1: citations improve the perceived accuracy of correct statements when compared with no explanations.

5.1.4 Hypotheses. In Experiment 1, all statements were correct. Participants tended to perceive the statements to be more accurate when presented with citations. We expected to have a similar observation for the correct statements in this experiment.

H2.1. Participants with explanations (citations or code) will report higher perceived accuracy for correct statements than those without explanations.

Kim et al. [20] showed that inconsistency highlighting in explanations for factual claims can reduce overreliance. Hence, we formulated these hypotheses:

H2.2. Participants with citations will report lower perceived accuracy on incorrect statements than those with code or no explanations, because inconsistency highlighting makes errors more salient.

H2.3. Participants with explanations (citations or code) will report lower perceived overall reliability than those without explanations.

H2.4. Following H1.3, participants with citations will report better system understanding than those with code or no explanations.

5.2 Results

From Figure 10, we again observed that for accurate statements, explanations (code or citations) tended to increase perceived accuracy compared with no explanations. Also, for incorrect statements, citations appear to reduce perceived accuracy when compared with both code and no explanations. Participants also tended to perceive the overall reliability to be lower and have a better system understanding with citations than with no explanations.

In the following, we provide the detailed analysis.

5.2.1 Perceived Accuracy of Correct Statements (H2.1 Supported). Participants in the citation condition rated the correct statements

the most accurate (M=2.79, SD=0.29), followed by the code condition (M=2.40, SD=0.72), and the no-explanation baseline (M=1.11, SD=0.95). A Kruskal-Wallis test indicated a significant main effect of explanation type on perceived accuracy of correct statements ($\chi^2(2) = 40.56$, p<.001). Post-hoc pairwise comparisons showed that both the citation condition (p<.001) and the code condition (p<.001) had significantly higher perceived accuracy than the baseline. The citation and code conditions did not differ significantly (p=.12).

5.2.2 Perceived Accuracy of Incorrect Statements (H2.2 Supported). Participants in the citation condition rated the incorrect statements the least accurate (M=−2.17, SD=1.15), followed by the code condition (M=0.03, SD=1.29), and the no-explanation condition (M=1.63, SD=1.05). A Kruskal-Wallis test indicated a significant main effect of explanation type on perceived accuracy ($\chi^2(2) = 57.17$, p<.001). The citation condition had significantly lower perceived accuracy than both the code condition (p<.001) and the no-explanation condition (p<.001). The code condition also had significantly lower perceived accuracy than the no-explanation condition (p<.001). Inconsistency highlighting in citations appears to be effective at surfacing errors, while code snippets and console outputs might require closer inspection to spot errors.

5.2.3 Perceived Overall Reliability (H2.3 Partially Supported). A Kruskal-Wallis test revealed a significant main effect of explanation type on perceived overall reliability ($\chi^2(2) = 8.74$, p=.013). Post-hoc pairwise comparisons showed that participants in the citation condition (M=0.77, SD=1.59) rated reliability significantly lower than those in the code condition (M=1.80, SD=0.81; p=.014), but not significantly lower than those without explanations (M=1.37, SD=0.96; p=.20). The difference between code and no explanation was also not significant (p=.13).

While Experiment 1 suggested that explanations improve perceived reliability, the open-ended responses indicated that error salience might offset the improvements. The offsetting effects may explain why neither explanation condition differed significantly from no explanations. A participant in the citation condition noted, "*The bot exhibits inconsistencies in basic calculations, rendering the analysis unreliable.*"

A stronger error salience with citations than with code might explain the lower perceived reliability with citations. Citations directly highlighted inconsistencies between stated facts and the underlying data, whereas code required participants to trace the logic themselves—and some could not detect the errors at all: "*The facts mostly matched the code outputs and calculations, so the insights appear accurate.*"

5.2.4 System Understanding (H2.4 Partially Supported). Participants in the code condition reported the highest system understanding (M=1.90, SD=0.99), followed by the citation condition (M=1.77, SD=1.79), and the baseline (M=0.97, SD=1.40). A Kruskal-Wallis test indicated a significant main effect of explanation type on system understanding ($\chi^2(2) = 9.93$, p=.007). Both the citation condition (p=.022) and the code condition (p=.022) had significantly higher system understanding than the no-explanation condition. However, the citation and code conditions did not differ significantly (p=.32). This contrasts with Experiment 1, where citations led to significantly higher understanding than code. Experiment 1 suggested that citations improve system understanding, likely because they make the analytic steps more accessible. With obvious errors, some participants in the citation condition were confused about how the system derives its answers: "*The steps themselves were very understandable on the right. It is therefore somewhat incomprehensible why some major errors occurred. I'm not sure what the AI did here.*" This erosion of confidence may have narrowed the gap between the citation and code conditions observed in Experiment 1.

6 Discussion

Here, we discuss how to extend our technique to support more analytic operations, the implications of our crowdsourced experiments for appropriate reliance, and the study limitations.

6.1 Technique Extensibility

While Section 3.1.1 focuses on the analytic operations in the application of clickstream analysis, our technique is highly extensible. It can be expanded to support a broader set of event-sequence analytic operations by following the procedure outlined in Section 3.1. First, we curate a new set of operations for the application (Sec. 3.1.1). Second, we classify the analytic operations into row, column, and filter operations (Sec. 3.1.2): operations for defining cohorts are row operations; operations for performing computations on cohorts are column operations; operations involving filtering of sequences or events are filter operations. Third, we implement the new operations as functions that can be invoked by the add, delete, and update tools (Sec. 3.1.3).

As an example, for outcome analysis, we may identify edge analysis [12] as a new analytic operation (step 1). Since this operation involves performing computations on a cohort, we classify this operation as a column operation (step 2). Finally, we can implement edge analysis as a function that takes a set of sequences as the input to calculate the odds ratio of each event attribute with respect to a specified outcome event as the output (step 3).

6.2 Citations and Appropriate Reliance

When evaluating an explanation technique, researchers often assess whether the explanation fosters appropriate reliance: users should trust the AI outputs when they are correct and reject them otherwise. Crowdsourced experiment 1 showed that participants tended to perceive correct data claims to be more accurate with citations when compared with no explanations. Crowdsourced experiment 2 indicated that participants appeared to perceive incorrect data claims to be less accurate with citations when compared with code and no explanations. Increasing perceived accuracy of correct claims while reducing perceived accuracy of incorrect ones is the premise of appropriate reliance.

Our crowdsourced experiments also echoed studies in domains other than data analysis. For example, Schemmer et al. [37] reported that explanations tend to increase reliance on AI advice. Kim et al. [20] suggested that inconsistencies in explanations for factual claims can reduce overreliance. Different from this body of work, we focused on studying a different domain of event sequence analysis. We further contributed a comparison between citations and

code as an explanation mechanism: for correct data claims, we do not have evidence that citations improve the perception of accuracy compared with code; for incorrect data claims, we found that citations reduced the perception of accuracy compared with code, presumably because the analytic steps and results in the floating window are more accessible than code.

6.3 Study Limitations

We conducted two crowdsourced experiments to examine how citations might shape users' assessment of event-sequence data claims. We observed that citations could foster appropriate reliance. However, we are prudent in drawing conclusions about the practical significance of this finding. Our crowdsourced experiments measured perceived accuracy and perceived reliability—not whether participants made better decisions. While perceived accuracy is a commonly used proxy [49, 50], a user who perceives a correct data claim as accurate may still fail to act on it, and a user who perceives an incorrect claim as inaccurate may not know how to correct it. Future work should investigate whether citations lead to better decision outcomes in more realistic analytic tasks.

7 Conclusion

In this paper, we introduced a technique linking event-sequence data claims to their analytic provenance in chatbots. To develop this technique, we identified a set of analytic operations for clickstream analysis, organized them through a pivot-table abstraction, and developed tools so an AI agent can construct a reference table for citing data claims within natural language responses. We demonstrated the technique in two applications: automated insight generation and post-hoc fact-checking. In both applications, users can inspect how a claim was derived by hovering over it rather than by reading code. Two crowdsourced experiments further suggested that citations could help foster appropriate reliance: they tended to raise the perceived accuracy of correct claims relative to no explanation and, with inconsistency highlighting, lowered the perceived accuracy of incorrect claims relative to both code and no explanation. These findings suggest that linking individual data claims to inspectable analytical steps and results can help users form assessments that are more sensitive to whether the available evidence supports or contradicts an AI-generated analysis.

References

[1] Sirwan Khalid Ahmed, Ribwar Arsalan Mohammed, Abdulqadir J. Nashwan, Radhwan Hussein Ibrahim, Araz Qadir Abdalla, Barzan Mohammed M. Ameen, and Renas Mohammed Khdhir. 2025. Using thematic analysis in qualitative research. *Journal of Medicine, Surgery, and Public Health* 6 (2025), 100198. doi:10.1016/j.glmedi.2025.100198

[2] Mubashara Akhtar, Oana Cocarascu, and Elena Simperl. 2023. Reading and Reasoning over Chart Images for Evidence-Based Automated Fact-Checking. In *Findings of the Association for Computational Linguistics: EACL 2023*. 399–414. doi:10.48550/arXiv.2301.11843

[3] Robert Amar, James Eagan, and John Stasko. 2005. Low-Level Components of Analytic Activity in Information Visualization. In *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization (INFOVIS '05)*. IEEE Computer Society, USA, 15. doi:10.1109/INFOVIS.2005.24

[4] Anthropic. 2026. Claude. https://claude.ai. Accessed: 2026-10-1.

[5] Serhat Aydin, Mert Karabacak, Victoria Vlachos, and Konstantinos Margetis. 2024. Large language models in patient education: a scoping review of applications in medicine. *Frontiers in Medicine* Volume 11 - 2024 (2024). doi:10.3389/fmed.2024.1477898

[6] Gagan Bansal, Tongshuang Wu, Joyce Zhou, Raymond Fok, Besmira Nushi, Ece Kamar, Marco Tulio Ribeiro, and Daniel Weld. 2021. Does the Whole Exceed Its Parts? The Effect of AI Explanations on Complementary Team Performance. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, New York. doi:10.1145/3411764.3445717

[7] Carrie J Cai, Samantha Winter, David Steiner, Lauren Wilcox, and Michael Terry. 2021. Onboarding Materials as Cross-functional Boundary Objects for Developing AI Assistants. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 43, 7 pages. doi:10.1145/3411763.3443435

[8] Fan Du, Ben Shneiderman, Catherine Plaisant, Sana Malik, and Adam Perer. 2017. Coping with Volume and Variety in Temporal Event Sequences: Strategies for Sharpening Analytic Focus. *IEEE Transactions on Visualization and Computer Graphics* 23, 6 (2017), 1636–1649. doi:10.1109/tvcg.2016.2539960

[9] Wafaa El Husseini. 2024. E-commerce Transactions + Clickstream. https://www.kaggle.com/datasets/wafaaelhusseini/e-commerce-transactions-clickstream. Kaggle dataset. Accessed: 2026-03-31.

[10] Yu Fu, Shunan Guo, Victor S. Bursztyn, Jane Hoffswell, Ryan Rossi, and John Stasko. 2024. "The Data Says Otherwise" – Towards Automated Fact-checking and Communication of Data Claims. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. ACM, New York. doi:10.1145/3654777.3676359

[11] Google. 2026. Google Gemini. https://gemini.google.com/app. Accessed: 2026-10-1.

[12] David Gotz and Harry Stavropoulos. 2014. DecisionFlow: Visual Analytics for High-Dimensional Temporal Event Sequence Data. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (Dec 2014), 1783–1792. doi:10.1109/TVCG.2014.2346682

[13] Shunan Guo, Fan Du, Sana Malik, Eunyee Koh, Sungchul Kim, Zhicheng Liu, Donghyun Kim, Hongyuan Zha, and Nan Cao. 2019. Visualizing Uncertainty and Alternatives in Event Sequence Predictions. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, 1–12. doi:10.1145/3290605.3300803

[14] Yi Guo, Shunan Guo, Zhuochen Jin, Smiti Kaul, David Gotz, and Nan Cao. 2022. Survey on Visual Analysis of Event Sequence Data. *IEEE Transactions on Visualization and Computer Graphics* 28, 12 (Dec. 2022), 5091–5112. doi:10.1109/TVCG.2021.3100413

[15] Gaole He, Abri Bharos, and Ujwal Gadiraju. 2024. To Err Is AI! Debugging as an Intervention to Facilitate Appropriate Reliance on AI Systems. In *Proceedings of the 35th ACM Conference on Hypertext and Social Media* (Poznan, Poland) *(HT '24)*. Association for Computing Machinery, New York, NY, USA, 98–105. doi:10.1145/3648188.3675130

[16] Gaole He, Stefan Buijsman, and Ujwal Gadiraju. 2023. How Stated Accuracy of an AI System and Analogies to Explain Accuracy Affect Human Reliance on the System. *Proc. ACM Hum.-Comput. Interact.* 7, CSCW2, Article 276 (Oct. 2023), 29 pages. doi:10.1145/3610067

[17] Saehan Jo, Immanuel Trummer, Weicheng Yu, Xuezhi Wang, Cong Yu, Daniel Liu, and Niyati Mehta. 2019. Verifying Text Summaries of Relational Data Sets. In *Proceedings of the 2019 International Conference on Management of Data* (Amsterdam, Netherlands) *(SIGMOD '19)*. Association for Computing Machinery, New York, NY, USA, 299–316. doi:10.1145/3299869.3300074

[18] Enkelejda Kasneci, Kathrin Sessler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, Stephan Krusche, Gitta Kutyniok, Tilman Michaeli, Claudia Nerdel, Jürgen Pfeffer, Oleksandra Poquet, Michael Sailer, Albrecht Schmidt, Tina Seidel, Matthias Stadler, Jochen Weller, Jochen Kuhn, and Gjergji Kasneci. 2023. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences* 103 (2023), 102274. doi:10.1016/j.lindif.2023.102274

[19] Majeed Kazemitabaar, Jack Williams, Ian Drosos, Tovi Grossman, Austin Z. 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*. ACM, New York. doi:10.1145/3654777.3676345

[20] Sunnie S. Y. Kim, Jennifer Wortman Vaughan, Q. Vera Liao, Tania Lombrozo, and Olga Russakovsky. 2025. Fostering Appropriate Reliance on Large Language Models: The Role of Explanations, Sources, and Inconsistencies. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. ACM, New York. doi:10.1145/3706598.3714020

[21] Yoonsu Kim, Jueon Lee, Seoyoung Kim, Jaehyuk Park, and Juho Kim. 2024. Understanding Users' Dissatisfaction with ChatGPT Responses: Types, Resolving Tactics, and the Effect of Knowledge Level. In *Proceedings of the 29th International Conference on Intelligent User Interfaces*. ACM, New York, 385–404. doi:10.1145/3640543.3645148

[22] Josua Krause, Adam Perer, and Harry Stavropoulos. 2016. Supporting Iterative Cohort Construction with Visual Temporal Queries. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (Jan. 2016), 91–100. doi:10.1109/tvcg.2015.

2467622

[23] Bum Chul Kwon, Min-Je Choi, Joanne Taery Kim, Edward Choi, Young Bin Kim, Soonwook Kwon, Jimeng Sun, and Jaegul Choo. 2019. RetainVis: Visual Analytics with Interpretable and Interactive Recurrent Neural Networks on Electronic Medical Records. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (Jan. 2019), 299–309. doi:10.1109/tvcg.2018.2865027

[24] Vivian Lai, Han Liu, and Chenhao Tan. 2020. "Why is 'Chicago' deceptive?" Towards Building Model-Driven Tutorials for Humans. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3313831.3376873

[25] Po-Ming Law, Zhicheng Liu, Sana Malik, and Rahul C. Basole. 2019. MAQUI: Interweaving Queries and Pattern Mining for Recursive Event Sequence Exploration. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (Jan 2019), 396–406. doi:10.1109/TVCG.2018.2864886

[26] John D. Lee and Katrina A. See. 2004. Trust in Automation: Designing for Appropriate Reliance. *Human Factors* 46, 1 (2004), 50–80. doi:10.1518/hfes.46.1.50.30392

[27] Yoonjoo Lee, Kihoon Son, Tae Soo Kim, Jisu Kim, John Joon Young Chung, Eytan Adar, and Juho Kim. 2024. One vs. Many: Comprehending Accurate Information from Multiple Erroneous and Inconsistent AI Generations. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency* (Rio de Janeiro, Brazil) *(FAccT '24)*. Association for Computing Machinery, New York, NY, USA, 2518–2531. doi:10.1145/3630106.3662681

[28] Q. Vera Liao and Jennifer Wortman Vaughan. 2023. AI Transparency in the Age of LLMs: A Human-Centered Research Roadmap. *arXiv preprint arXiv:2306.01941* (2023). doi:10.48550/arXiv.2306.01941

[29] Nelson Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating Verifiability in Generative Search Engines. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics, Singapore, 7001–7025. doi:10.18653/v1/2023.findings-emnlp.467

[30] Zhicheng Liu, Bernard Kerr, Mira Dontcheva, Justin Grover, Matthew Hoffman, and Alan Wilson. 2017. CoreFlow: Extracting and Visualizing Branching Patterns from Event Sequences. *Computer Graphics Forum* 36, 3 (2017), 527–538. doi:10.1111/cgf.13208

[31] Zhicheng Liu, Yang Wang, Mira Dontcheva, Matthew Hoffman, Seth Walker, and Alan Wilson. 2017. Patterns and Sequences: Interactive Exploration of Clickstreams to Understand Common Visitor Paths. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (Jan. 2017), 321–330. doi:10.1109/tvcg.2016.2598797

[32] Sana Malik, Fan Du, Megan Monroe, Eberechukwu Onukwugha, Catherine Plaisant, and Ben Shneiderman. 2015. Cohort Comparison of Event Sequences with Balanced Integration of Visual Analytics and Statistics. In *Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI15)*. ACM, 38–49. doi:10.1145/2678025.2701407

[33] Megan Monroe, Rongjian Lan, Hanseung Lee, Catherine Plaisant, and Ben Shneiderman. 2013. Temporal Event Sequence Simplification. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (Dec 2013), 2227–2236. doi:10.1109/TVCG.2013.200

[34] OpenAI. 2026. ChatGPT. https://chatgpt.com. Accessed: 2026-10-1.

[35] Jian Pei, Jiawei Han, B. Mortazavi-Asl, H. Pinto, Qiming Chen, U. Dayal, and Mei-Chun Hsu. 2001. PrefixSpan,: mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings 17th International Conference on Data Engineering*. 215–224. doi:10.1109/ICDE.2001.914830

[36] Virgilijus Sakalauskas and Dalia Kriksciuniene. 2024. Personalized Advertising in E-Commerce: Using Clickstream Data to Target High-Value Customers. *Algorithms* 17 (2024), 27. doi:10.3390/a17010027

[37] Max Schemmer, Niklas Kühl, Cass Benz, Andrea Bartos, Gerhard Satzger, and Mahima Pushkarna. 2023. Appropriate Reliance on AI Advice: Conceptualization and the Effect of Explanations. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*. ACM, New York. doi:10.1145/3581641.3584066

[38] Timo Schick, Jane Dwivedi-Yu, Roberto Dessí, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: language models can teach themselves to use tools. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '23)*. Curran Associates Inc., Red Hook, NY, USA, Article 2997, 13 pages.

[39] Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get Your Vitamin C! Robust Fact Verification with Contrastive Evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 624–643. doi:10.48550/arXiv.2103.08541

[40] Jaideep Srivastava, Robert Cooley, Mukund Deshpande, and Pang-Ning Tan. 2000. Web usage mining: discovery and applications of usage patterns from Web data. *SIGKDD Explor. Newsl.* 1, 2 (Jan. 2000), 12–23. doi:10.1145/846183.846188

[41] Chris Stolte, Diane Tang, and Pat Hanrahan. 2008. Polaris: a system for query, analysis, and visualization of multidimensional databases. *Commun. ACM* 51, 11 (Nov. 2008), 75–84. doi:10.1145/1400214.1400234

[42] Katerina Vrotsou and Aida Nordman. 2019. Exploratory Visual Sequence Mining Based on Pattern-Growth. *IEEE Transactions on Visualization and Computer Graphics* 25, 8 (Aug. 2019), 2597–2610. doi:10.1109/tvcg.2018.2848247

[43] David Wadden, Kyle Lo, Lucy Lu Wang, Arman Cohan, Iz Beltagy, and Hannaneh Hajishirzi. 2022. MultiVerS: Improving scientific claim verification with weak supervision and full-document context. In *Findings of the Association for Computational Linguistics: NAACL 2022*, Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (Eds.). Association for Computational Linguistics, Seattle, United States, 61–76. doi:10.18653/v1/2022.findings-naacl.6

[44] Aileen P. Wright, Adam T. Wright, Allison B. McCoy, and Dean F. Sittig. 2015. The use of sequential pattern mining to predict next prescribed medications. *Journal of Biomedical Informatics* 53 (2015), 73–80. doi:10.1016/j.jbi.2014.09.003

[45] Yang Wu, Yao Wan, Mennatallah El-Assady, and April Yi Wang. 2026. StepMIND: A Visual Framework for Stepwise, Multimodal, and Bidirectional Explanations of AI-Generated Data Analysis Pipeline. In *Proceedings of the 31st International Conference on Intelligent User Interfaces (IUI '26)*. ACM, 522–539. doi:10.1145/3742413.3789070

[46] Liwenhan Xie, Chengbo Zheng, Haijun Xia, Huamin Qu, and Zhu-Tian Chen. 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*. ACM, New York. doi:10.1145/3654777.3676374

[47] Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. 2023. A Critical Evaluation of Evaluations for Long-form Question Answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Toronto, Canada, 3225–3245. doi:10.18653/v1/2023.acl-long.181

[48] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*. doi:10.48550/arXiv.2210.03629

[49] Ming Yin, Jennifer Wortman Vaughan, and Hanna Wallach. 2019. Understanding the Effect of Accuracy on Trust in Machine Learning Models. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. doi:10.1145/3290605.3300509

[50] Yunfeng Zhang, Q. Vera Liao, and Rachel K. E. Bellamy. 2020. Effect of confidence and explanation on accuracy and trust calibration in AI-assisted decision making. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (Barcelona, Spain) *(FAT* '20)*. Association for Computing Machinery, New York, NY, USA, 295–305. doi:10.1145/3351095.3372852

[51] Zhicheng Zhou, Jing Li, Suming Qiu, Junjie Huang, Linyuan Qiu, and Zhijie Sun. 2025. DeepJSONEval: Benchmarking Complex Nested JSON Data Mining for Large Language Models. *arXiv preprint arXiv:2509.25922* (2025). doi:10.48550/arXiv.2509.25922

[52] Kazi Tasnim Zinat, Saimadhav Naga Sakhamuri, Aaron Sun Chen, and Zhicheng Liu. 2025. A Multi-Level Task Framework for Event Sequence Analysis. *IEEE Transactions on Visualization and Computer Graphics* 31, 1 (Jan. 2025), 842–852. doi:10.1109/tvcg.2024.3456510

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009