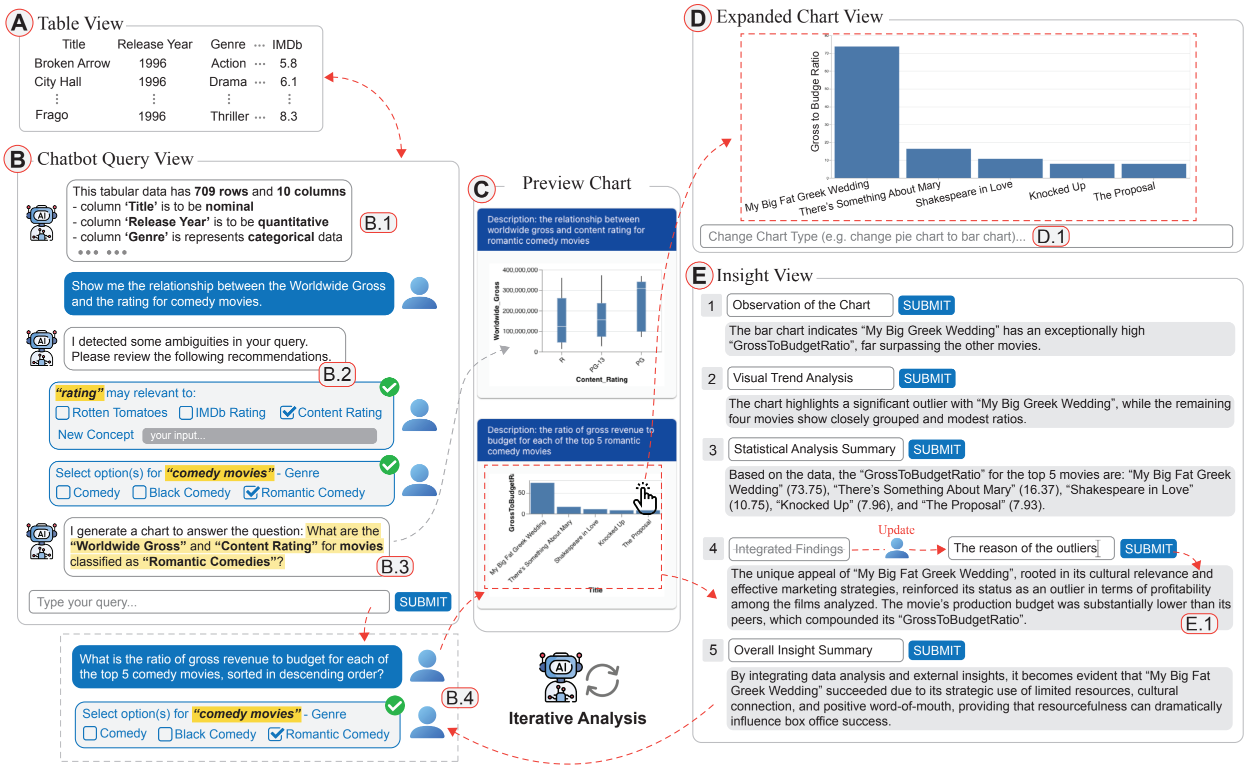
From Questions to Insights: A Natural Language Interface for Transforming Utterances into Insights in Exploratory Data Analysis

Shichen Gao, Zezheng Sun, Fan Lei, Lin Li, Yuyan Bao, and Jieqiong Zhao



This tabular data has **709 rows** and **10 columns**
- column **'Title'** is to be **nominal**
- column **'Release Year'** is to be **quantitative**
- column **'Genre'** is represents **categorical** data

Show me the relationship between the Worldwide Gross and the rating for comedy movies.

I detected some ambiguities in your query.
Please review the following recommendations.

"rating" may relevant to:
☐ Rotten Tomatoes ☐ IMDb Rating ☑ Content Rating
New Concept: your input...

Select option(s) for **"comedy movies"** - Genre
☐ Comedy ☐ Black Comedy ☑ Romantic Comedy

I generate a chart to answer the question: What are the **"Worldwide Gross"** and **"Content Rating"** for **movies** classified as **"Romantic Comedies"**?

Type your query... SUBMIT

What is the ratio of gross revenue to budget for each of the top 5 comedy movies, sorted in descending order?

Select option(s) for **"comedy movies"** - Genre
☐ Comedy ☐ Black Comedy ☑ Romantic Comedy

Description: the relationship between worldwide gross and content rating for romantic comedy movies

Description: the ratio of gross revenue to budget for each of the top 5 romantic comedy movies

1 Observation of the Chart SUBMIT
The bar chart indicates "My Big Greek Wedding" has an exceptionally high "GrossToBudgetRatio", far surpassing the other movies.

2 Visual Trend Analysis SUBMIT
The chart highlights a significant outlier with "My Big Greek Wedding", while the remaining four movies show closely grouped and modest ratios.

3 Statistical Analysis Summary SUBMIT
Based on the data, the "GrossToBudgetRatio" for the top 5 movies are: "My Big Fat Greek Wedding" (73.75), "There's Something About Mary" (16.37), "Shakespeare in Love" (10.75), "Knocked Up" (7.96), and "The Proposal" (7.93).

4 ~~Integrated Findings~~ The reason of the outliers SUBMIT
The unique appeal of "My Big Fat Greek Wedding", rooted in its cultural relevance and effective marketing strategies, reinforced its status as an outlier in terms of profitability among the films analyzed. The movie's production budget was substantially lower than its peers, which compounded its "GrossToBudgetRatio".

5 Overall Insight Summary SUBMIT
By integrating data analysis and external insights, it becomes evident that "My Big Fat Greek Wedding" succeeded due to its strategic use of limited resources, cultural connection, and positive word-of-mouth, providing that resourcefulness can dramatically influence box office success.

Fig. 1: QtoINS contains five views to display data, charts, and insights: (A) Table View, displays the uploaded tabular data, including column headers and row entries; (B) Chatbot Query View, enables people to ask analytical questions in natural language and interact with a chatbot to retrieve target data; (C) Preview Chart View, presents a gallery of thumbnail charts automatically generated based on human queries; (D) Expanded Chart View, shows an enlarged interactive version of a selected chart coded by Vega-Lite specifications, supporting interactions such as hover, pan, and zoom; and (E) Insight View, provides autogenerated data insights based on the selected chart, leveraging chain-of-thought reasoning.

Abstract—Large language models (LLMs) have made obtaining answers through natural language queries more accessible and efficient; however, general-purpose LLMs, primarily trained on textual data, often struggle to interpret structured tabular data, resulting in plausible but factually questionable data insights by filling in missing details with invented information. Despite benchmark datasets and fine-tuned models tailored for exploratory data analysis, customized LLMs may frequently misrepresent people's intent, highlighting the need for interactive validation mechanisms combining automated checking with human-in-the-loop clarification. We introduce QtoINS, a conversational chatbot system that identifies ambiguities in human utterances and proactively seeks clarification inputs. QtoINS decomposes the questions-to-insights generation process into three core components: (1) natural language queries into target data, (2) target data into visual charts, and (3) visual charts into insights. The decomposition enables human intervention at each stage of the analytical process. The effectiveness of QtoINS was evaluated through system evaluation and user feedback.

Index Terms—Large language model (LLM), SQL, chart generation, insights, exploratory data analysis

◆

1 INTRODUCTION

Recent advances in Large Language Models (LLMs) have significantly simplified interactions between humans and AI-assisted systems, making daily tasks, such as getting simple instructional assistance (e.g., "How do data visualization and visual analytics differ?"), convenient and efficient [22]. However, despite their impressive performance in conversational scenarios, current general-purpose LLMs encounter notable challenges in performing analytical tasks that data analysts

---

- *Shichen Gao, Lin Li, Yuyan Bao, and Jieqiong Zhao are with Augusta University, Augusta, GA, USA. E-mail: {shgao, lli1, yubao, jiezhao}@augusta.edu.*
- *Zezheng Sun is with University of Wisconsin-Madison, Madison, WI, USA. E-mail: zsun359@wisc.edu.*
- *Fan Lei is with Arizona State University, Tempe, AZ, USA. E-mail: flei5@asu.edu*
- *Correspondences to Yuyan Bao and Jieqiong Zhao. E-mail: {yubao, jiezhao}@augusta.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

routinely undertake [21]. Tasks such as querying “What was the total sales amount for March?” or requesting to “Plot monthly sales figures for the past year,” require accurate interpretation of structured tabular data and precise analytical execution, where general-purpose LLMs often struggle.

One potential way to partially overcome this limitation is by enabling LLMs to translate natural language analytical queries into executable Python code, given the data schema of a relational database [93]. While technically feasible, this approach assumes a significant degree of programming knowledge from those who perform the task. For individuals with limited coding expertise, interpreting Python code and outputs can rapidly shift from an analytical experience to a cumbersome debugging and reading task. As a result, what should ideally be an accessible analytical tool becomes a daunting code interpretation task, reducing the effectiveness and usability of AI-assisted analytical reasoning. For instance, Kandel et al. [23] highlighted that many non-professional programmers frequently encounter data analysis tasks, often stuck in debugging activities rather than smoothly analyzing and interpreting data. Therefore, reducing technical barriers and enabling non-professionals to efficiently analyze and interpret data remains an essential challenge for the design of analytical systems that leverage LLMs technology. We aim to design a data analysis tool tailored for *casual experts* [41], who have substantial domain knowledge but are not extensively trained in programming or visual design principles. An accessible tool powered by LLMs can support such analysts by lowering technical barriers and guiding the analytical process through a conversational interface, making data exploration more inclusive for a broader audience [52].

The visual analytics community has made significant contributions to support data analysis using human queries: some researchers have developed systems that use LLMs to transform natural language queries into charts [12, 56, 72], while others have created tools that leverage LLMs to automatically generate insights by interpreting charts [31, 69]. However, even fine-tuned LLMs may frequently misrepresent human intentions or generate plausible but questionable insights without warning. These challenges stem mainly from *unresolved ambiguities* within human queries and *inherent variability* in LLMs-generated output. LLMs often yield inconsistent results across multiple runs (e.g., identifying different data columns for the same query), further undermining their practical usage for analytical tasks. For instance, in data analysis, subtle ambiguities (e.g., an unclear selection of desired categories or data range) can lead to biased conclusions when ambiguities remain unresolved. These limitations underscore a critical need to enhance people’s awareness of potential misalignment between their intent (often expressed through underspecified utterances) and the desired data, while being aware of the reliability issues in LLM-generated responses.

Driven by the need to validate and refine the LLM-generated outputs through feedback [49] and inspired by the visual analytics literature emphasizing tight integration between humans and machines [55], we resolve ambiguities in human queries and improve data insight extraction by integrating automatic validation mechanisms with human-in-the-loop clarification. Building on the established pipeline of natural language interfaces (NLIs) for exploratory data analysis [55], we decompose the questions-to-insights process into three components: (1) translating natural language queries into target data, (2) transforming target data into charts, and (3) deriving insights from these charts. For each component, we employ prompt engineering techniques guiding LLMs to generate outputs aligned with human intent. All results will be publicly available [anonymized link].

Our main contributions include the following.

- A conversational chatbot system, *QtoINS*, comprising structured pipelines to transform natural language into target data, subsequently generating charts, and deriving insights. Each step integrates automatic validation and human refinement to increase awareness of potential inconsistencies in responses generated by LLMs.
- We design a novel LLMs-assisted interactive approach to resolve *concept and value ambiguities* in human utterances for a relational data table, ensuring target data aligns with human intent. Then, we fine-tune an LLM with human queries and extracted features on target data to automatically generate charts (i.e., Vega-Lite specifications).
- *Chain-of-thought (CoT)* reasoning is employed to interpret the generated charts and derive data insights. The use of CoT reasoning enables human analysts to iteratively refine these insights in a step-by-step manner and perform web searches to supplement the data with external knowledge.
- A *system evaluation* is conducted to assess the effectiveness of each component, a design study with Computer Science faculty and staff is conducted to refine the interface and functionality, and a comparative usability study with casual experts is conducted to evaluate the usability of QtoINS vs. ChatGPT for data analysis.

2 RELATED WORK

Conversational systems for data analysis span a broad spectrum of interconnected domains, with three areas particularly shaping the foundation of our approach: resolving ambiguity in queries, translating natural language into visualizations, and generating insights from charts through automated reasoning.

2.1 Ambiguity Detection in Natural Language Queries for Tabular Data

Interactive data analysis systems typically translate natural language queries into SQL statements to retrieve data from relational databases. Compared to manually writing SQL queries for extracting data from Tabular Data, this approach eliminates the need for data analysts to understand SQL language, making it more accessible.

Earlier works primarily focus on understanding the semantics of natural language and often assume that human queries are unambiguous. For example, several studies [3, 57, 84] adopt *rule-based approaches* that rely on language parsers to translate natural language into SQL. While effective for well-defined inputs, these approaches do not scale well to diverse datasets or complex query structures. *Deep learning-based approaches* [18, 76, 86, 87] demonstrate better scalability with the support of benchmark datasets such as WikiSQL [95], Spider [88], and Spider 2.0 [27]. However, these methods often perform poorly on unseen data, as they typically require large amounts of training data for each new scenario. Recently, *transformers and large language models*, such as BERT [11], T5 [54], GPT-3 [7], and GPT-4 [47], have shown strong capabilities in capturing rich semantic and syntactic features by training on large-scale corpora.

More recent works [16,42,45,56,63,68,77,78] have been focusing on detecting and resolving ambiguities in human queries to ensure accurate query translation, due to the inherent ambiguity of natural language [15]. **In this work, we classify ambiguities into two categories: *ambiguous concepts*, referring to terms in queries that do not match the column names, and *ambiguous values*, referring to those in queries that do not match the table content (details in Section 3.1).** To the best of our knowledge, no prior LLM-based work has addressed the problem of ambiguous values in human queries; our approach is the first to tackle this challenge. The following gives an overview of related work on detecting and resolving ambiguous concepts in human queries.

Wang et al. [75] uses a weakly supervised model based on BERT [11] to output alerts if ambiguous concepts are detected. However, its performance is limited by BERT’s fixed architecture and heavy task-specific training. AMBROSIA [60] and [5] leverage LLMs to generate multiple candidate SQL statements from queries that contain ambiguous concepts. Applying those approaches [5, 60] to interactive data analysis would require data analysts to possess sufficient SQL knowledge and to discern subtleties among the candidate SQL statements, yet without any guarantees of accuracy, as they do not refer to the data values to validate the candidate statements. Sphinteract [91] relies on data analysts to identify ambiguous concepts, and forwards LLM generated SQL statements to data analysts iteratively until a satisfactory statement is confirmed. [53]’s work uses the beam search method to generate a set of candidate SQL statements with associated probabilities for queries involving ambiguous concepts, and then leverages the probability distribution to identify and clarify potential ambiguities in the SQL for human review. Like ours, their approach avoids SQL knowledge requirements, but differs in that our candidates are directly extracted from the data schema of the given table.

Different from prior works, we developed an LLM-assisted, interactive, step-wise approach that resolves concept and value ambiguities by guiding data analysts to locally validate the intended concepts and values from the candidates, which are extracted from the data schema of the given table and the table content, respectively. Our approach requires no knowledge of SQL and reduces the risk of privacy leakage.

2.2 Natural Language Interface for Data Visualization

There are three major gaps in natural language interface for data visualization. To start with, extant studies may overlook the distribution of feature values (data characteristics in Section 3.2), meaning that previous approaches largely fail to account for how values within each feature are actually distributed. For example, *rule-based approaches* such as Mackinlay's foundational work [39], Show Me [40], Voyager [81], and Draco [43], relied on design heuristics and logical constraints to suggest visualizations. Although effective in structured contexts, these approaches struggled to scale and adapt to complex datasets, where feature distribution plays a critical role. More recently, the advent of *LLMs* (e.g., GPT-3 [7], LLaMA [73], GPT-4 [47]) has enabled more semantically rich and flexible recommendation systems. Frameworks such as [30], LIDA [12], and Data Formulator [77, 78] harness LLMs to generate visualizations directly from abstract natural language or tabular prompts, but they too typically abstract away the underlying value distributions. This highlights an importance gap: the need to account for data characteristics, such as data type and its inherent challenges, for instance, string data type could introduce ambiguity due to the nature of human natural languages, as elaborated in Section 2.1.

Moreover, previous research has provided only limited support for the specific analytical *intent* of data analysts, leading to misalignment. For *rule-based approaches*, challenges arise when handling complex or underspecified inputs, especially when human intent is difficult to infer. For instance, initial systems such as DataTone [16], Eviza [63], and FlowSense [85] employed grammar-driven and rule-based parsers to address ambiguity in human queries. Emerging systems like NL4DV [45] used structured parsing to map queries to Vega-Lite specifications. This is a similar case for *ML-based approaches*, such as Deepeye [35], VizML [20], Data2Vis [13], and DashBot [10], which introduced data-driven learning to predict visualization design choices. However, both of these approaches lacked a nuanced understanding of human intent and showed limited generalization capabilities across tasks. For *transformer- and LLM-based approaches*, most rely on an end-to-end generation approach, making it difficult for data analysts to identify which steps (data mapping, chart generation, insight extraction) cause issues if the final results are problematic. Specifically, neural architecture approaches trained on large datasets such as nvBench [34, 36] and ncNet [37] have emerged, leveraging transformer models to translate queries into valid declarative specifications (e.g., Vega-Lite) for data visualization. chartgpt [72] [1] fine-tuned the T5 model, utilizing LLMs to enhance query translation to charts. However, because these methods treat visualization recommendation as a single black-box process, they provide little visibility into intermediate steps, a gap our research aims to address.

Lastly, there exists a research gap in the current benchmark datasets. Extant studies often rely on rigid column specifications and explicit visualization instructions [8, 36], such as specifying the x-axis as a temporal column and the y-axis as total revenue for a line chart. Such approaches make the mapping between data columns and visualization overly explicit, which limits the capacity of natural language interfaces to perform meaningful query-to-chart mapping. However, in real-world scenarios, casual experts typically express their needs in natural language queries rather than through detailed, step-by-step visualization instructions. To address this gap, our study emphasizes capturing the essence of human queries rather than procedural requests, and enabling the natural language interface to infer query-to-chart mappings. Specifically, we modified the existing benchmark dataset [8], which maps tabular data to visual charts, by introducing human queries tailored for casual experts to lower the technical barrier of conversational interfaces.

2.3 Data Insights Generation from Charts

The process of generating data insights from charts has been highlighted as a key component of data-driven decision-making [48, 55]. Oftentimes, individuals seek patterns, trends, and anomalies in information visualization tools that can inform their choices [1]. Data insights often reflect both analytic knowledge derived from the dataset and domain knowledge brought by the analyst, and they are produced through interpretation of visualizations, not the visualizations themselves [4]. The integration of text explanations within charts has been shown to improve both clarity and comprehension for readers [70]. Consequently, researchers are increasingly interested in integrating natural language descriptions to enhance data analysis and interpretation. Before LLMs, visualization tools such as Voder [67], GeoExplainer [28], and GROOT [17], relied on template-based methods, which provided interactive data facts, spatial modeling explanations, and user-editable templates, respectively. However, reliance on predefined rules sometimes reduced flexibility and made adaptation less straightforward.

As the capabilities of LLMs in generating data narratives and insights continue to evolve, researchers have investigated how LLMs might enrich visualization frameworks. Building on this momentum, several systems have integrated LLMs: LightVA [93] helps analysts break down high-level analytical goals into actionable tasks and gain insight; LEVA [94] supports multiple stages of visual analysis and enables retracing of analytical history; and InsightLens [80] employs an agent-based framework to automatically record, organize, and navigate insights. These systems were developed before recent advances in reasoning strategies such as CoT prompting [79], and therefore place less emphasis on making LLMs' reasoning processes adjustable or transparent to people. CoT improves LLM performance on complex tasks by generating intermediate reasoning steps [79]. Our approach employs CoT reasoning to structure and explain the insights derived from charts, improving both transparency and depth of insights.

We design a novel interaction method that allows analysts to update and validate insights more effectively (details in Section 3.3). Our approach adopts CoT reasoning to enable automated insight extraction through a structured five-step process: chart observation, visual trend analysis, statistical analysis, synthesis of findings, and comprehensive summarization (shown in Figure 1(E)). This process aligns with the design guidelines proposed by Stokes et al. [70]. In the second step, visual trend analysis, our guiding prompt integrates key features as summarized in prior work [1, 59, 62]. To support human-in-the-loop insight editing, our system enables people to regenerate insights based on feedback instructions.

We conducted an online study comparing insights generated with CoT prompting to those produced by the GPT-4o API. Later, a comparative usability study with casual experts evaluated QtoINS against ChatGPT for insight generation. Results show that ChatGPT often hallucinates when interpreting charts, consistent with recent findings [19], as commercial LLMs often rely on pre-trained knowledge rather than interpreting chart semantics, leading to the introduction of external or irrelevant information. In contrast, our CoT-based approach provides improved accuracy, transparency, and higher-quality insights.

3 METHOD

QtoINS is a novel LLMs-assisted interactive system that supports an iterative complete analytical process – from interpreting a human query to deriving insights – as shown in Figure 2. This section introduces its three components: ①, ② and ③ respectively.

3.1 Human Query to Target Data

Figure 3 shows that QtoINS resolves ambiguities in a query, eventually translating the query into the validated SQL statement retrieving data

---

[1] To avoid confusion with ChatGPT, we adopt the lowercase form chartgpt in this paper.

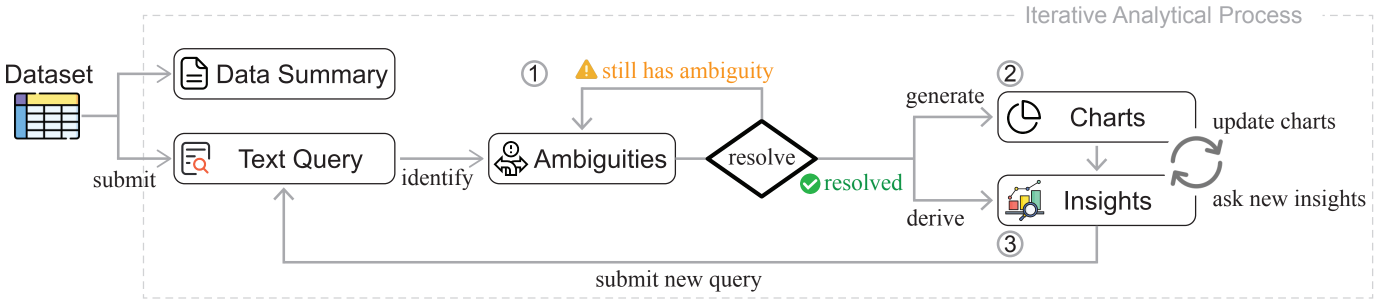


Fig. 2: QtoINS supports a complete workflow: ① interatively resolving ambiguities in a human query, and translating the query into target data with human intent; ② generating charts with the human query and target data; ③ deriving insights with the human query, target data, and the charts. Analysts can iteratively refine the generated insights by entering natural language commands, and start a new iteration of analysis by entering a follow-up query with the insights gained from the previous iterations.

for chart generation. The solid and dashed arrows indicate the back-and-forth flow of information transmitted among a human, QtoINS, and LLMs. We use the movie dataset in Figure 1 to illustrate this procedure, and assume QtoINS receives the dataset with column names. including "Title", "Rotten Tomatoes Rating", "IMDB Rating", "Content Rating", "Release Year" and "Genre".

Problems As shown in Figure 3, the process starts with a query from a human, e.g., "Compare the rating for comedy movies in recent years". It mentions terms that do not match the column names in the dataset, i.e., "rating". We refer to such terms as *ambiguous concepts* (abbreviated as `Amb-C` in the figure).

In addition, the data type of column "Genre" is a String, which is the source of ambiguous values. In this example, after inspecting the dataset, we found that the data stored in the column includes "Comedy", "Black Comedy" and "Romantic Comedy". It is unclear whether the user intends to include all those types when referring to "comedy movies". Thus, we consider "comedy movies" as an *ambiguous value* (abbreviated to `Amb-V` in the figure), and "Recent Year" as another example of numerical values.

Without guiding examples, LLMs can generate SQL statements for the above query. The following shows one: [2]

```
SELECT 'Release Year', AVG('IMDB Rating'),
AVG('Rotten Tomatoes Rating')
FROM movies
WHERE Genre= 'Comedy' AND 'Release Year'≥2020;
```

Note that the generated SQL statement may vary across different runs. We observe that LLMs translates "rating" to a predicted one like the above, and "recent years" to "released year after ×× year", where ×× can be arbitrary, e.g., 2020 in the above, and "comedy movies" to "movies" whose genre is 'Comedy'.

Our Approach Our approach identifies a fragment of SQL statements sufficient to express common *single-table queries for exploratory data analysis (EDA)*. Observing that tabular data are organized into rows and columns (with each cell uniquely determined by a row-column combination), and EDA over tabular data is mainly directed by two operations: selection (filtering data away) and aggregation (aggregating many records, often via grouping) [44]. Accordingly, our fragment includes the **SELECT**, **WHERE**, and **GROUP BY** clauses, together with standard aggregate functions, such as **MIN**, **MAX**, **AVG**, **SUM**, and **COUNT**. Note that, for a syntactically correct SQL statement, the SELECT and GROUP BY clauses and the arguments of aggregate functions must refer to column names. In addition, inspired by SQLf [6], we allow { } wrapped texts to appear in our SQL statement to express ambiguities. To facilitate locating ambiguous concepts and values in SQL Stmt, we further require that, for any binary expression occurring in the **WHERE** clause, the left operand corresponds to a column header and the right operand corresponds to a value. For example, the expression "Genre = Comedy" is syntactically valid, whereas "Comedy = Genre" is not. These constraints are incorporated into the prompt's guiding examples (Appendix A.1.2), and QtoINS verifies that each SQL Stmt satisfies them through syntactical check.

Locating Ambiguities. In this setting, all the wrapped texts in SQL Stmt is considered as ambiguous. QtoINS further examines unwrapped text, occurring in column positions; if such a text does not match any column name, it is wrapped – thereby marking it as ambiguous – for subsequent ambiguity resolution. Ambiguous values are located similarly. QtoINS marks unwrapped text occurring in value positions as ambiguous if matched values are not found with respect to the data values in a given table.

Resolving Ambiguities. We use a two-step interactive procedure to refine SQL Stmt and synthesize human *validated* SQL statements by leveraging LLMs. Each step resolves one type of ambiguities, highlighted in blue and yellow in Figure 3, respectively.

Step 1: Resolve Ambiguous Concepts The following shows a generated SQL Stmt, where "rating" is identified as an ambiguous concept.

```
SELECT Title,{rating}
FROM movies
WHERE Genre= Comedy AND 'Release Year'≥recent years;
```

As shown in Figure 3, for each of the `Amb-C`$_n$, i.e., `Amb-C`$_i$, QtoINS guides LLMs (Appendix A.1.3) to predict candidate column names, and prompts dialogues for people to confirm them in Chatbot Query View (Section 4.2). Figure 4 shows the dialogues for the user to confirm all occurrences of `Amb-C`$_n$. The dialogues are designed with appropriate widgets with respect to data types. For example, multiple checkboxes are provided for selecting relevant ratings; an input textbox is provided for defining a new numeric concept, as highlighted in Figure 4. In this case, QtoINS will further validate the new concept with LLMs (Appendix A.1.4), due to the free-form nature of texts. See details in Section 4.2.

Step 2: Resolve Ambiguous Values With resolved `Amb-C`, resolving ambiguous values requires us to gather the relevant data for a given concept. For numeric data, e.g., "Release Year", a slider allows the people to select its range.

Gathering the relevant data for "comedy movies" for column "Genre" is challenging. Performing string-matching algorithms for this data-gathering task is not satisfying, as we need to consider the semantics of "comedy movies". Directly delegating the task to LLMs may lead to significant candidates missing, as LLMs does not guarantee to scan all data rows. Deep learning algorithms lack the learning capability of LLMs in recommending relevant data in the context of human utterances.

Thus, QtoINS firstly employs MPNet [66] for semantic search to rank candidate phrases with a maximum of N distinct samples from the dataset, where N can be set in a global configuration file. MPNet is widely used for semantic textual similarity tasks and semantic search,

---

[2] The generated SQL statement also includes **GROUP BY** and **ORDER BY**, which are irrelevant to our discussion and thus are omitted.

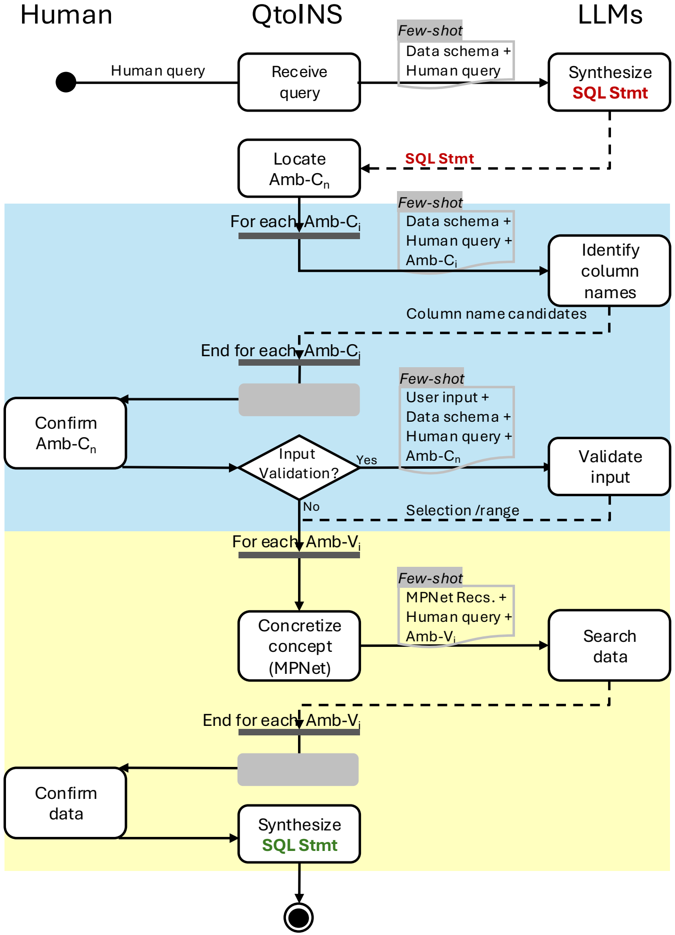


Fig. 3: QtoINS resolves ambiguous concepts (`Amb-C`) and values (`Amb-V`) in LLMs generated SQL Stmt from a human query through a two-step interactive procedure that synthesizes validated SQL Stmt with human intent.

and outperforms traditional models like BERT [11] in capturing fine-grained semantic similarities.

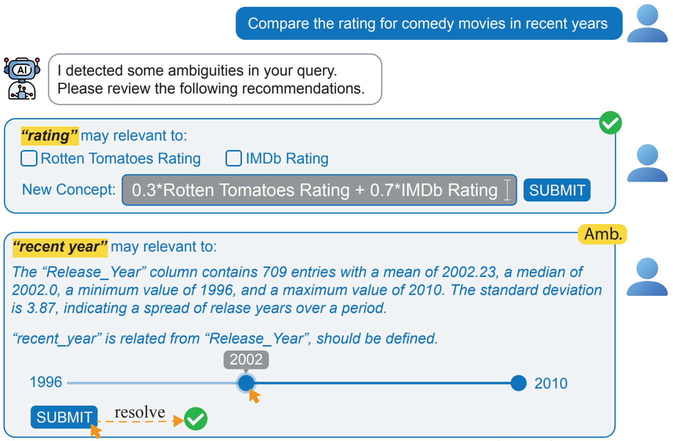
Then, QtoINS selects the top M candidates (recommended by MP-Net) that are semantically aligned with "Comedy", and subsequently delegates the LLMs (Appendix A.1.5) to predict the relevant data candidates with the human query as shown in Figure 3 (bottom). The number of candidates M is configurable as well. In this example, LLMs predicates exactly all three relevant data values in the dataset, and further prompts dialogues for the user's confirmation in Chatbot Query View.

Finally, QtoINS synthesizes *validated* SQL Stmt with the confirmed data values, as shown below:

```
SELECT Title,'IMDB Rating','Rotten Tomatoes Rating'
FROM movies
WHERE Genre In ('Comedy','Black Comedy',
                'Romantic Comedy')
      AND 'Release Year' between 2002 and 2010
```

3.2 Human Query and Target Data to Charts

This component takes target data, and generates interactive charts aligned with human intent. QtoINS fine-tunes GPT-4o [3] with both human queries and the data distribution. Table 1 summarizes the techniques employed by systems for chart generation. Our approach is inspired by [25], who showed that the effectiveness of a scatterplot



Compare the rating for comedy movies in recent years

I detected some ambiguities in your query.
Please review the following recommendations.

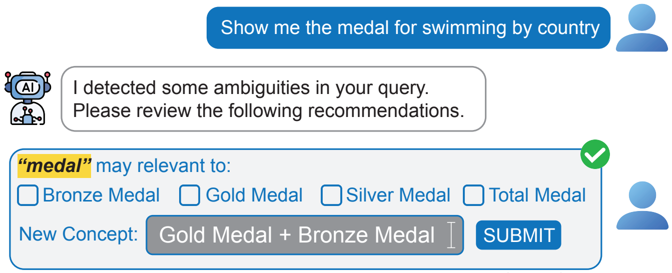
"rating" may relevant to:
☐ Rotten Tomatoes Rating ☐ IMDb Rating
New Concept: 0.3*Rotten Tomatoes Rating + 0.7*IMDb Rating

"recent year" may relevant to:

The "Release_Year" column contains 709 entries with a mean of 2002.23, a median of 2002.0, a minimum value of 1996, and a maximum value of 2010. The standard deviation is 3.87, indicating a spread of relase years over a period.

"recent_year" is related from "Release_Year", should be defined.

Fig. 4: Dynamic concept creation based on user input for defining a linear combination of two ratings.



Show me the medal for swimming by country

I detected some ambiguities in your query.
Please review the following recommendations.

"medal" may relevant to:
☐ Bronze Medal ☐ Gold Medal ☐ Silver Medal ☐ Total Medal
New Concept: Gold Medal + Bronze Medal

Fig. 5: Dynamic concept creation based on user input for defining medal categories.

depends on both the analytical task and the underlying data distribution. Following this observation, we incorporate both human queries and *data characteristics* to guide chart generation (see Section 5.2 for evaluation comparison). To the best of our knowledge, we are the first to fine-tune LLMs by incorporating data characteristics for chart generation based on human queries.

Feature Extraction on Target Data As shown in Figure 6Ⓐ, we use the feature representation method described in VizML [20], which transforms data into 841 features, including the number of columns and column types. As a result, each data sample in our fine-tuning dataset consists of one human query, 841 feature values of the target data, and the first r rows of the data, where $r \leq 3$, since LLM-generated content can be error-prone when too much content is produced with the corresponding Vega-Lite specifications.

Fine-tuning Strategy We focus on fine-tuning the GPT-4o model to effectively learn embeddings of Vega-Lite specifications from structured data representations of target data. Since generating complex content inherently increases the likelihood of errors [83], we aim to minimize the length of the GPT-4o outputs during fine-tuning, especially when the target data itself occupies a substantial portion of the response. To achieve this, our approach generates partial Vega-Lite specifications that are structurally aligned with the target data but limited in scope. Specifically, the charts are rendered using only the first three rows of the dataset (or fewer, if the dataset contains fewer). After inference, we manually replace the truncated data with target data to complete the visualization. This controlled fine-tuning strategy enhances the accuracy, consistency, and reliability of the generated visual specifications, while reducing error propagation during fine-tuning.

Fine-tuning Dataset Our fine-tuning dataset is adapted from the VisEval benchmark dataset [8], a state-of-the-art collection designed to map human queries to charts and encompassing a broad variety of chart types. The benchmark dataset includes pairs of human queries and corresponding Vega-Lite specifications [61], spanning seven types of charts, including Pie, Bar, Stacked Bar, Line, Grouping Scatter, Scatter, and Grouping Line.

---

[3] GPT-4o can be accessed at https://platform.openai.com/docs/models/gpt-4o.

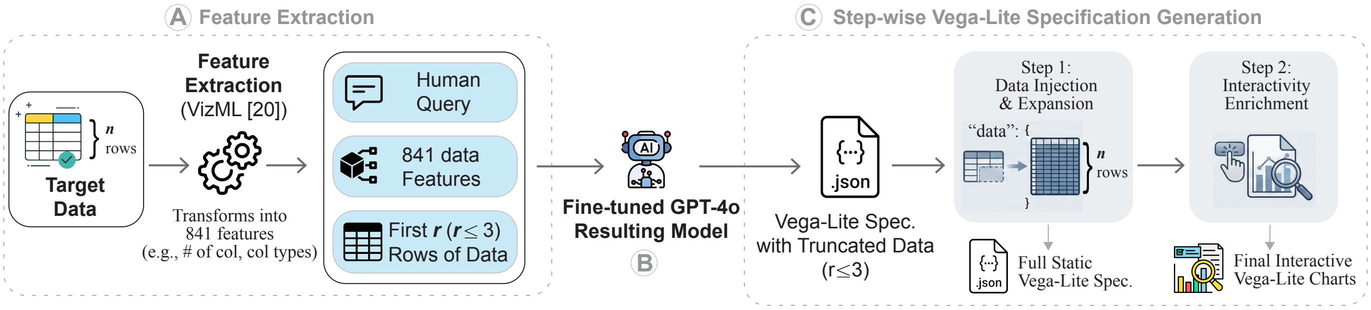

Ⓐ Feature Extraction
Ⓒ Step-wise Vega-Lite Specification Generation
Target Data
n rows
Feature Extraction (VizML [20])
Transforms into 841 features (e.g., # of col, col types)
Human Query
841 data Features
First r (r≤ 3) Rows of Data
Fine-tuned GPT-4o Resulting Model
Ⓑ
AI
{...} .json
Vega-Lite Spec. with Truncated Data (r≤3)
Step 1: Data Injection & Expansion
"data": {
n rows
}
Full Static Vega-Lite Spec.
Step 2: Interactivity Enrichment
Final Interactive Vega-Lite Charts

Fig. 6: QtoINS generates charts from target data by using fine-tuned GPT-4o.

Table 1: List of sample systems using different techniques for chart generation. The second and third columns indicate whether the system uses a human query or extracted features from target data in the chart generation process.

System	Human Query	Extracted Features	Method
NL4DV-LLM [56]	Yes	No	Crafted LLM prompts using NL4DV to generate charts [45].
chartgpt [72]	Yes	No	Fine-tuned a T5 model to generate charts.
ncNet [37]	Yes	No	Combined human queries and data schema using a transformer-based sequence-to-sequence model to generate charts.
NL4DV [45]	Yes	No	Used a rule-based method to interpret human queries and generate charts.
VizML [20]	No	Yes	Used a neural network with three hidden layers to predict charts, taking only 841 data features as input.
Setlur et al. [64]	Yes	No	Parsed human queries based on the syntactic and semantic constraints of analytical expressions to generate charts.

To make the queries more representative of real-world, casual experts usage, we manually revised 70 queries (10 for each chart type). For example, the original query "List the number of games in each season with a line chart, shown by the x-axis in ascending" was changed to "List the number of games in each season," shifting the emphasis from explicit visualization instructions to the underlying analytical intent. In addition, we systematically updated the Vega-Lite specifications in the VisEval benchmark dataset [8] to ensure compatibility with the latest version of Vega-Lite. Details of the fine-tuning prompt are provided in Appendix A.2.

The 70 queries are split into a training set and a validation set at an 8:2 ratio, resulting in 56 samples for fine-tuning and 14 for validation. For each chart type, we use 8 training samples and 2 validation samples. We set the batch size to 4, the learning rate multiplier to 1, and trained the model for 5 epochs. OpenAI cloud fine-tuning time is 26 minutes and costs $85.50.

Our final fine-tuning set consists of 56 samples, which satisfies the recommended starting point of 50 well-crafted samples for effective fine-tuning of GPT-4o. [4] This aligns with recent findings that commercial LLMs (e.g., OpenAI, Gemini) achieve the best performance when fine-tuned with approximately 50-150 unique examples [82], further supporting the sufficiency of our fine-tuning dataset size. An ablation test and a detailed empirical comparison of accuracy against other systems are described in Section 5.2, which further justifies that our setup can achieve effective fine-tuning with a small fine-tuning set at a modest computational cost.

Resulting Model After fine-tuning, we obtain a GPT-4o-based model capable of generating charts combining human queries and data distributions (Figure 6Ⓑ). Our final results (Figure 7) achieved a training loss of 0.1228 and a validation loss of 0.1320, indicating satisfying performance and strong generalization capabilities. The model parameters are stored on OpenAI's platform, and can be accessed via a specific model API key.

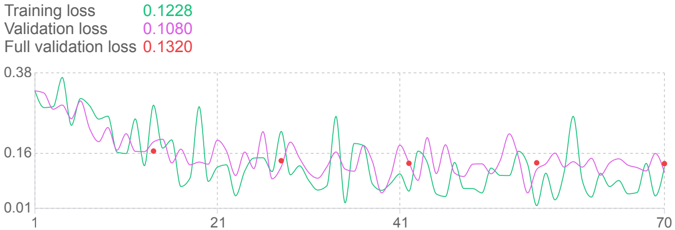


Fig. 7: Screenshot of training and validation loss during fine-tuning using the GPT-4o cloud service. The x-axis indicates the number of training steps (70 steps over 5 epochs). The model exhibits stable convergence, with training loss decreasing consistently and validation loss remaining consistently low.

Step-wise Vega-Lite Specification Generation This module (Figure 6Ⓒ) is maintained locally by our system, without any involvement of LLMs. As shown in the figure, our fine-tuned model outputs the Vega-Lite specification with respect to the given r rows of data. This specification is subsequently substituted by the complete target dataset locally, yielding a full specification of the static chart. Furthermore, we deliberately add interaction primitives into the Vega-Lite specification locally since interaction is integral to EDA.

3.3 Charts to Insights

This component takes the human query, target data, and chart (static rendered image exported from Vega-Lite specification), and implements CoT reasoning to generate the data insights in natural languages from charts by leveraging LLMs. QtoINS delegates LLMs to perform data analysis on target data by using the OpenAI Responses API. [5] Invoking the API instructs LLMs to perform exploratory analysis, as well as generating five structured insights (Appendix A.3.1), following the design guidelines advocated by [1, 33]. The five default insights are generated according to the following instructions: ① Observation of the Chart: describing the overall chart type, axes, and encodings; ② Visual Trend Analysis: identifying trends, outliers, clusters, or distributions using patterns commonly studied in visualization literature; ③ Statistical Analysis Summary: inferring statistical properties, such as averages, variations, or correlations; ④ Integrated Findings: connecting multiple observations to build a coherent understanding; ⑤ Overall Insight Summary: generating a concise, high-level takeaway of the chart. In particular, the insight ② integrates known visual features to support sensemaking, including the identification of notable temporal trends, outliers, group comparisons, and data clusters.

An example of five default insights for the query 'Show me the relationship between the Worldwide Gross and the rating for comedy movies," is shown in Figure 1(E). Each insight includes a pair of

---

[4] https://platform.openai.com/docs/guides/supervised-fine-tuning

[5] https://platform.openai.com/docs/quickstart?api-mode=responses

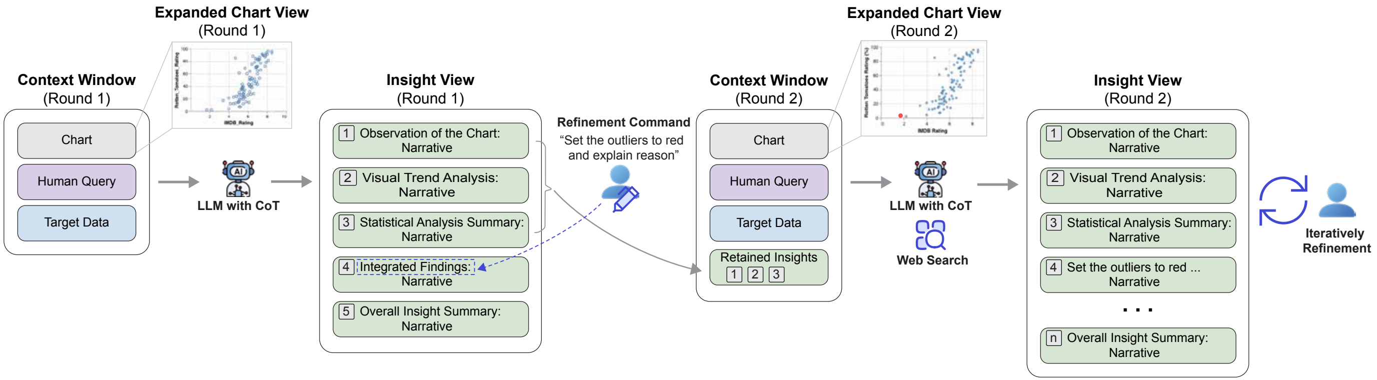

Expanded Chart View
(Round 1)
Context Window
(Round 1)
Chart
Human Query
Target Data
LLM with CoT
Insight View
(Round 1)
1 Observation of the Chart: Narrative
2 Visual Trend Analysis: Narrative
3 Statistical Analysis Summary: Narrative
4 Integrated Findings: Narrative
5 Overall Insight Summary: Narrative
Refinement Command
"Set the outliers to red and explain reason"
Expanded Chart View
(Round 2)
Context Window
(Round 2)
Chart
Human Query
Target Data
Retained Insights
1 2 3
LLM with CoT
Web Search
Insight View
(Round 2)
1 Observation of the Chart: Narrative
2 Visual Trend Analysis: Narrative
3 Statistical Analysis Summary: Narrative
4 Set the outliers to red ... Narrative
. . .
n Overall Insight Summary: Narrative
Iteratively Refinement

Fig. 8: Our Charts-to-Insights component generates default 5 structured insights from human query, target data, and chart by CoT reasoning. People can then iteratively refine these insights by editing the insight actions to produce updated insights; web search is enabled only during the refinement stage.

(i) an *action* summary, which casual experts can edit to reflect their analytical intent, and (ii) a detailed *narrative*, which a LLM generates to elaborate, justify, and contextualize the action. As illustrated in Figure 8 round 2, casual experts can iteratively refine insights in Insight View (Section 4.5) by entering natural language commands via input boxes of actions. For example, if a casual expert submits the command "Set the outlier to red and explain reason" within insight ④, QtoINS instructs LLMs (see Appendix A.3.2 for its guiding examples) to generate new insights with its web-search functionality enabled. [6] Specifically, in the incremental updates, the current context window includes (i) retained insights (ii) current chart, (iii) current query, and (iv) target data. A similar process is performed as illustrated in Round 2 of Figure 8, where QtoINS packages the current context window and prompts the LLMs to regenerate the insights while preserving the retained insights; the chart is updated only when the natural language command requires it.

To clarify, regenerated insights do not necessarily follow the default five-action structure. Because natural language commands are open-ended, we intentionally provide flexibility to accommodate diverse analysis requests. In our insight refinement guiding prompt, we instruct the LLMs to perform CoT reasoning, and the final insight is always an overall insight summary. Only the insight refinement process involves web search.

4 USER INTERFACE

QtoINS is a web-based system supporting exploratory data analysis through the questions-to-insights process for casual experts [41]. The system is built using React and Vega-Lite for the front end and Flask for the back end. This section describes the five views, using the movie dataset as a running example.

4.1 Table View

Tabular visualization is a common feature in visual analytics dashboards, and provides precise representations that support the detailed exploration of raw data. Table View can be involved in different stages of data analysis. At the beginning, Table View enables people to upload tabular data files and inspect the detailed contents. In later stages of the analysis, people can validate autogenerated insights by inspecting raw values (Figure 1(A)).

When people submit queries in the Chatbot Query View (Section 4.2, Figure 1(B)), our system detects potential ambiguities and suggests clarification. If needed, people can refer to the raw data table. This interplay between high-level queries and low-level data inspection makes Table View a handy component. However, examining raw data in a large dataset is cumbersome. The focus of QtoINS is to reduce reliance on manual table inspection and instead support casual experts in performing smooth, analytical analysis.

4.2 Chatbot Query View

Chatbot Query View is the core component for people to steer the analysis, and resolve the automatically detected ambiguities in human queries through interactive dialogues, combining the reasoning capabilities of LLMs with human-in-the-loop validation. The validation process increases people's awareness of the unreliability issue of LLMs; however, we observed that casual experts tried to ask questions in different ways to verify the accuracy of insight. This is exactly what the data analysis or sensemaking process could be; they may like to verify an insight from different perspectives to test their hypotheses [24]. Once a tabular dataset is uploaded, the chatbot displays a summary textbox (Figure 1(B.1)). If a data type is categorical, distinct categories will be listed.

Within Chatbot Query View, people can resolve ambiguities in their queries one by one through dialogues, e.g., Figure 1(B.2), which are designed with appropriate widgets against data types. Categorical attributes are specified via checkboxes, while numerical attributes are specified via slider bars. QtoINS also allows people to define new concepts through input boxes (Figures 4 and 5), enabling people to create derived fields on-the-fly. For example, in Figure 5, people can define a new concept (i.e., column), such as the sum of `Gold Medal` and `Bronze Medal`. If a division operation is detected in the definition, QtoINS initiates a safe arithmetic check, preventing division by zero errors. Once a concept is defined, it will be incorporated into subsequent chart generation, supporting more adaptive analysis.

Once all ambiguities are resolved, the chatbot presents a clarified query used for chart generation. The thumbnails of autogenerated charts are appended in Preview Chart View (Section 4.3, Figure 1(C)), where each chart is linked to its originating clarified query. These links are saved for historical tracking, enabling users to revisit or improve previous analyses.

Chatbot Query View maintains a dynamic conversational thread, displaying context-aware prompts, system-generated recommendations, and human selections. It supports real-time modification during the clarified query formulation process. The tight integration of LLM semantic reasoning, system-guided clarification, and human-in-the-loop validation enables Chatbot Query View to promote transparency and facilitate effective collaboration. This workflow ultimately yields human-verified SQL statements.

4.3 Preview Chart View

Exploratory data analysis often involves examining multiple charts before discovering insightful findings. To support this iterative process, Preview Chart View (Figure 1(C)) maintains a history of autogenerated visual charts in response to clarified human queries derived from Chatbot Query View. Each thumbnail of a chart is accompanied by a brief textual description summarizing its content and analytical intent at the top. This layout allows people to revisit previously generated visualizations. When people select a chart of interest, it is opened in Expanded Chart View (Figure 1(D)) for detailed examination and interaction.

---

[6] OpenAI Responses API (with `external_web_access=true`).

Preview Chart View serves as an overview or history tracking role by organizing visual charts in an accessible format, promoting effective comparison, repeated revisits, and improved analytical engagement.

4.4 Expanded Chart View

Expanded Chart View combines interaction with adaptive visual representations, enabling users to iteratively refine data encodings and generate nuanced insights. Selecting a thumbnail from Preview Chart View opens a larger, detailed chart rendered with Vega-Lite, which supports interactions such as hover tooltips, zooming, and panning for close inspection (e.g., outliers). Users can also export data and charts for reporting.

As different chart types reveal distinct patterns or trends, Expanded Chart View provides a natural language input box to change the visual encoding (e.g., chart types, color schemes, axis variables). This customization feature complements automatic chart generation, allowing users-even without visualization or programming expertise-to refine representations through simple text commands.

4.5 Insight View

Insight View supports casual experts in deriving meaningful insights from charts, which is particularly helpful for less experienced people who may need to spend time understanding chart elements. In addition, the work by Stokes et al. [70] has shown that the presence of textual insights along with univariate line charts can significantly aid interpretation. Thus, Insight View, which provides initial autogenerated, step-by-step explanations, can help people interpret selected charts and discover new insights.

When a chart is selected from Preview Chart View, its insights are displayed in Insight View (Figure 1(E)) using a CoT reasoning approach. Each CoT insight is structured into two parts per step: (1) an action that summarizes the reasoning step, and (2) an autogenerated explanation produced by an LLM that elaborates on the corresponding observation or inference. To enrich contextual understanding and support more profound reasoning, our system may also perform web searches in the background (e.g., retrieving relevant facts, definitions, or domain-specific references) to supplement the generated explanations. People can edit the action directly, and the system delegates the explanation generation to the LLM, which updates the explanation accordingly.

Insight View also supports human-in-the-loop insight refinement, allowing people to issue natural language commands to regenerate or refine an insight explanation by editing the corresponding action. An edit triggers updates only for subsequent steps, preserving earlier content and reducing cognitive load. If the system determines that adjusting the visual encoding would benefit interpretability for the modified insights (e.g., highlighting outliers in a different color), it updates the visualization in both Preview Chart View and Expanded Chart View to align with the updated insight.

People can copy the insight content for follow-up queries in Chatbot Query View. This mechanism is especially useful when the focus of analysis shifts (e.g., changing target data or analytical goals). In addition, Preview Chart View also supports bookmarking of charts and associated insights. Together, these features support a flexible workflow that empowers people to engage in iterative, human-centered data exploration, and refinement of insights.

Design Highlights We believe that our innovations in the chatbot system shift the traditional paradigm of dashboard interaction in three aspects. (1) Structured and modular interaction pipeline. We designed a structured and modular pipeline with three components, each interactively controllable via natural language instructions. This modular approach enables reuse of intermediate results, mitigates the "lost in the middle" issue common in commercial LLMs, and allows independent module upgrades. Our system breaks down complex data analysis tasks into smaller, manageable steps (see Section 9.1 for further discussion). Unlike commercial LLMs and end-to-end chart generation systems (e.g., ChartGPT) that regenerate all outputs upon any input change, our approach improves controllability and transparency. (2) Human-in-the-loop ambiguity resolution. In Chatbot Query View, casual experts are engaged to resolve ambiguities via a human-in-the-loop mechanism rather than relying solely on LLM outputs. Our system solicits input through customized widgets to improve the accuracy of downstream generation of charts and insights. Our ambiguity resolver allows casual experts to dynamically define new concepts on-the-fly using domain knowledge. We acknowledge that we are not the first to use human-in-the-loop to support the generation of new (ambiguous) concepts using LLMs. For example, Data Formulator [77, 78] supports more sophisticated data transformation functions (e.g., cumulative sums) and enables experienced data scientists to inspect LLMs generated code, whereas we focus on supporting casual experts through simple arithmetic operations. Our contributions focus on human-in-the-loop ambiguous value resolution: consistent with prior text-to-SQL work (e.g., Spider [88]), they treat predicting the correct SQL structure and column selection (i.e., ambiguous concept resolution in our context) as the primary challenge, while expecting condition values to be verified and finalized through user interaction via an interface. (3) Editable and selectively updatable outputs. Both Expanded Chart View and Insights View support editable and selectively updatable outputs. Expanded Chart View allows casual experts to adjust visual elements (e.g., bar width, point size) without triggering insight regeneration when changes are stylistic, and Insights View enables editing or selective updating of insights, retaining satisfactory ones. This granular update mechanism promotes transparency and efficiency.

5 SYSTEM EVALUATION

We evaluated the effectiveness of our system by independently evaluating its three primary components: (1) ambiguity resolution, (2) chart generation, and (3) insight derivation.

For the first two components, we used the NLV benchmark dataset [68], which provides 30 representative cases (24 single-chart and 6 multi-chart) and corresponding 814 natural language queries collected by an online user study. These queries were from online participants who are public audiences, and reflect authentic linguistic variation, ambiguity, and underspecification. The dataset is well-suited for our evaluation as it includes ambiguities, and spans ten chart types to assess support for diverse visualization tasks and visual literacy. We compare the output of our system for these two components against existing benchmark models and NLIs. These comparisons help quantify the improvements introduced by our questions-to-insights process and LLMs-assisted ambiguity detection and chart generation.

For the final component, we compared 50 data insights generated with CoT prompting to GPT-4o API outputs in a crowdsourcing study. Subjective ratings indicate that the CoT-based approach yielded significantly better perceived insights.

As shown in Appendix E Table 11, GPT-4o was chosen for the system evaluation because it is a fast, intelligent, and flexible GPT model [46]. The detailed reasons are listed below. To start with, GPT-4o supprots fine-tuning, which is essential to our system, further details are explained in Appendix E Table 12. Furthermore, none of the listed models use an internal reasoning process (i.e., no reasoning tokens), making our chain-of-thought reasoning approach particularly valuable. While GPT-4o is not the cheapest option, it offers higher speed than both GPT-3.5 Turbo and GPT-4o Mini, which is an important factor for user interaction, and great intelligence than those two models. Although GPT-4.5 is more intelligent, GPT-4o is significantly more affordable, costing roughly 1/30th for input and 1/15th for output tokens. Lastly, its max output tokens are sufficient for our study. As for the models released after our study, such as GPT-o4 and GPT-5, they are understandably more advanced. Nevertheless, we believe the trade-offs between available models at the time of the study still made GPT-4o an excellent choice. More importantly, QtoINS adopts a loosely coupled design that enables each component to plug in any LLMs for future research experiments.

5.1 Human Query to Target Data

We evaluated our system (with GPT-4o API as the backbone model) on its ability to extract targeted data from natural language queries. Comparisons were made against two baselines: ChatGPT-4.5, one of the most advanced LLMs at the time of writing, and NL4DV-LLM

Table 2: Sample human queries for ambiguous concepts and values detection among our system, ChatGPT-4.5, and NL4DV-LLM [56].

Ambiguity Type	Sample Human Query	Our System	ChatGPT -4.5	NL4DV-LLM (GPT-4)
Ambiguous Concepts	SQ1: Show me rating for action movies.	✓	✓	✓
	SQ2: Show me medal for ice hockey by country.	✓	✓	✓
Ambiguous Values	SQ3: Show me the relationship between calories and total fat in beef sandwiches.	✓	×	×
	SQ4: Show me total medals for water sports by country.	✓	×	×
	SQ5: Plot the average horsepower vs. year for all volkswagens.	✓	×	×
	SQ6: Show me total medals for skating by country.	✓	×	×

(with GPT-4 API as the backbone model) [56]. Our goal is to assess system's human intent interpretation, ambiguity resolution, and data retriever for downstream visualization. Ambiguities and representative examples are summarized in Table 2.

Evaluation Dataset. We compiled a dataset of 50 natural language queries for this evaluation (see supplement), including: (1) 30 questions corresponding to the 30 cases in the NLV benchmark dataset [68], and (2) 20 additional queries that we created and manually annotated. Among these 50 queries, three contain ambiguous concepts, and eight contain ambiguous values.

Evaluation Workflow All three systems received the required inputs (a human query and a data file) to ensure smooth operation, and the evaluation was based on their output data. For ChatGPT-4.5, we verified whether the retrieved data matched the expected results by inspecting the generated Python code. For NL4DV-LLM, we used the Python package released on GitHub repository [7] and evaluated whether the data embedded in the generated Vega-Lite specifications correctly reflected the human intent. For our system, we assessed the synthesized SQL statements by checking whether these statements correctly retrieved the desired data.

Evaluation Criteria We evaluated each system based on whether it could produce the target data or a subset of target data implied by a human query.

- For queries that contain ambiguous concepts (e.g., SQ1 in Table 2), a system is considered correct if it selects any reasonable rating-related column (e.g., IMDB rating, Rotten Tomatoes rating, or both).
- For queries that contain ambiguous values (e.g., SQ6 in Table 2), a system is considered wrong if it fails to detect any relevant categories in the dataset (e.g., Speed Skating, Figure Skating, Short Track Skating).

Results and Special Cases Our system achieved 98% accuracy, outperforming ChatGPT-4.5 (84%) and NL4DV-LLM (70%). For the queries listed in Table 2, all three systems can detect ambiguous concepts; however, ChatGPT-4.5 and NL4DV failed in all human queries containing ambiguous values.

We observed two noteworthy patterns in the responses of ChatGPT-4.5:

- **Defining new concept:** While ChatGPT-4.5 can reasonably resolve ambiguous concepts, it guesses human intent based on the query. For example, in response to SQ1 in Table 2, ChatGPT-4.5 selects both `IMDB rating` and `Rotten Tomato rating`. However, for the query "Show me the overall rating in action movies," ChatGPT-4.5 calculates the *average* values of `IMDB rating` and `Rotten Tomato rating`. Our system prompts the two columns that are related to rating, and people can define the *overall rating* on-the-fly as: $0.3 \times$ `IMDB rating` $+ 0.7 \times$ `Rotten Tomato rating` (Figure 4).
- **Resolving ambiguous value:** We found that ChatGPT-4.5 selects the data by guessing possible values within the dataset, rather than scanning the values in a data column. For example, in a sandwiches dataset, [8] when people asked SQ3 in Table 2, ChatGPT-4.5 only selected data labeled as "beef" sandwiches. However, through precise data filtering, our system can identify both "beef" and "steak" sandwiches.

5.2 Human Query and Target Data to Charts

Ablation Test We focus on component 2 (charts generation), shown in Figure 2, to conduct an ablation study due to the inseparability of component 1, which is explained in detail in Section 9.4. Specifically, we applied the same training strategy to separately fine-tune a feature-only model and a question-only model for generating Vega-Lite specifications. The GPT-4o model without fine-tuning is also included in the accuracy comparison using the same input for benchmarking purposes. We used 70 samples from the 1,150-sample VisEval dataset as our training and validation sets, and the remaining 1,080 samples were used as the test set. All samples have been manually crafted to sound more like queries from a casual expert (by removing step-by-step visualization instructions that specify chart types or explicitly map the x- and y-axes). Our ablation test was conducted manually by running all test data across four models. The reason for manual evaluation is that we do not judge correctness based on the original fixed chart type, because the original human queries explicitly requested chart types, whereas our crafted queries did not. As long as the Vega-Lite chart generated from the human query and the target data satisfies the casual expert's needs, we count it as correct. For example, even if the original annotation required a bar chart, a pie chart would still be counted as correct as long as it meets the query intent.

The statistical results (see Table 3) show that our model achieves the highest or near-highest accuracy across most chart types, demonstrating strong generalization. It performs particularly well on most popular chart types such as bar and pie charts, reaching accuracies above 0.99. Although performance drops slightly on more complex or less frequent chart types (such as grouping scatter and grouping line) the model remains competitive compared to the baselines. Overall, our method attains the best overall accuracy (0.9769), outperforming models that rely solely on data features, human queries, or no fine-tuning. This indicates the effectiveness of combining both data features and human query signals with fine-tuning. The ablation study shows that 50 samples are sufficient for fine-tuning: despite the small dataset, accuracy improves by 3.6%.

Table 3: Accuracy comparison across 7 chart types for our model, fine-tuned model using only data features, fine-tuned model using only human query, and GPT-4o without fine-tuning.

Chart Type	**Count**	**Ours**	**Only Data Features**	**Only Human Query**	**Without Fine-tuning**
Bar	707	**0.9929**	0.9943	0.9844	0.9675
Pie	235	**1.0000**	0.9957	0.9957	0.9574
Line	45	0.9111	0.9556	**1.0000**	0.7778
Scatter	24	0.8750	0.8750	**0.9167**	0.8333
Stacked Bar	36	0.8333	**0.8889**	0.7222	0.8056
Grouping Line	11	0.8182	**0.9091**	0.8182	0.6364
Grouping Scatter	22	**0.7727**	0.5000	0.7273	0.7273
Overall	1080	**0.9769**	0.9759	0.9704	0.9407

Baseline Comparison We evaluated our system, NL4DV-LLM, and chartgpt (with Llama 3 as the backbone model), using the 24 single-chart cases from the NLV dataset and the 24 single-chart cases we used in Task 1 for our comparative usability study, for a total of 48 chart cases. NL4DV-LLM and chartgpt are both natural language to visualization tools (no insight generation). chartgpt and NL4DV-LLM both can produce multiple candidate visualizations from human queries. However, neither system proactively resolves ambiguities in the human query; in contrast, QtoINS actively prompts for clarification when it detects an ambiguous concept or an ambiguous value. Moreover, chartgpt and NL4DV-LLM adopt an end-to-end generation strategy, whereas our

---

[7] https://github.com/nl4dv/nl4dv/tree/master

[8] https://www.kaggle.com/datasets/davinm/subway-restaurant-nutrition-data

system uses a modular architecture that enables independent control of each component. During our baseline comparison, the input consisted of a human query and the tabular data file. The output was a Vega-Lite specification that represents the resulting chart. A generated chart was considered correct if it accurately fulfilled the intent expressed in the human query. Additionally, for queries that contain ambiguities, if our system was able to detect and clarify these ambiguities through its clarification process, and the resulting chart matched the clarified intent, it was also considered correct. Based on this evaluation criterion, NL4DV-LLM, chartgpt and our system achieved accuracies of 83.3%, 72.9%, and 97.9% respectively. A full breakdown of the results is provided in the supplement.

5.3 Charts to Insights

We conducted an online experiment on Prolific to compare insights generated by our CoT-based approach with those generated by the GPT-4o API. Both methods received the same inputs: a human query, a data file, and a chart. Each participant completed 10 trials; in each trial, they viewed a chart at the top of the screen and two insights presented side-by-side below it (the left-right order was randomized). The 10 trials were randomly drawn from a pool of 50 charts, comprising 24 single charts from the NLV benchmark dataset [68] and 26 randomly selected from the fine-tuning dataset in Section 3.2.

Both datasets use generally understood domains (e.g., cars, movies, superstore sales) and common chart types seen in everyday visualizations (e.g., news reports, stock market charts), making them suitable for a broad participant pool. As a result, participants' judgments primarily reflect the perceived quality of the insights rather than domain expertise. Accordingly, Prolific participants (age > 18, location US only) are competent for this evaluation. In addition to Likert ratings, we collected free-text justifications to support and validate participants' choices. The NLV paper [68] also notes in their future work the possibility of broadening the participant pool using crowdsourcing platforms.

To ensure fairness, we removed any visual highlights from both sets of insights but preserved formatting such as bullet points to encourage careful reading (layout shown in Appendix G). Participants rated each insight on a 5-point Likert scale and provided justifications for their ratings. The study involved 20 participants, yielding 400 ratings (2 insights × 10 trials × 20 participants), with an average completion time of 28 minutes and an average payment of $6.50.

As shown in Figure 9, CoT-generated insights ($\mu = 4.19$, $\sigma = 0.96$) were rated significantly higher than GPT-4o API-generated insights ($\mu = 3.65$, $\sigma = 1.12$), with a paired-sample t-Test confirming the difference ($p < .0001$). We categorized participants' reasons for rating the insights generated by CoT and GPT-4o API, the results are shown in Tables 4 and 5, respectively. Among all responses, 86.8% of the responses for CoT were positive, with common feedback highlighting that the insights are detailed and clear, the chart description is well-written, and the conclusion is useful. However, 11.2% of the responses mentioned that CoT insights were wordy. In contrast, 65% of the responses for GPT-4o API are positive, the most frequent feedback being that the insights are easy to understand. Meanwhile, 34.6% of the responses are negative, mainly criticizing the conclusion of a chart being too simple.

6 DESIGN STUDY

To refine our system prior to the evaluation with casual experts (Section 7), we conducted pair analytics sessions [2] with seven Computer Science faculty and staff to solicit suggestions. This section outlines the participants, session procedures, and key feedback.

Participants As listed in Appendix C Table 7, all participants regularly perform data analysis tasks and are familiar with LLMs, but have limited experience with interactive data visualization tools that support dynamic chart generation. Their top three data analysis tools are Excel, Python, and Matlab, and none of them had prior exposure to visual analytics platforms such as Tableau. Our user feedback session was conducted in a lab environment equipped with a 30-inch monitor to provide a suitable setting for interacting with a dashboard system with multi-coordinated views.

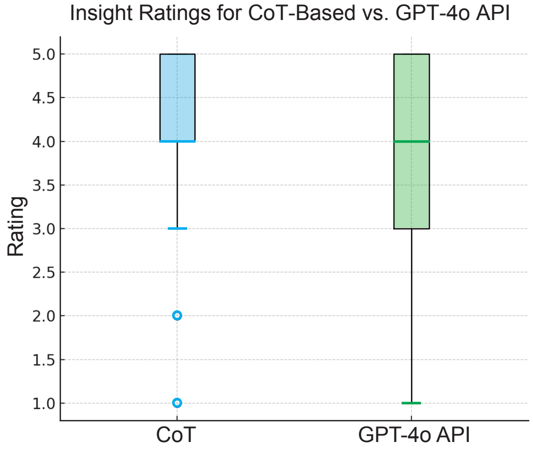


Fig. 9: Comparison of participant ratings for insights generated by the CoT-based approach and the GPT-4o API. CoT-based insights received significantly higher ratings ($p < .0001$, paired-sample t-Test).

Table 4: Supportive and opposing reasons for CoT-generated insights, together with the percent and accumulations.

	Reason	Pct.(%)	Accum.
Support	The insight is detailed and clear	34.8	34.8
	The description of the chart is well-written	24.0	58.8
	The conclusion is useful	15.7	74.5
	The structure of the insight is organized	5.9	80.4
	The insight includes both statistical summaries and visual analyses, which makes it convincing	5.4	85.8
	The insight seems more like it was written by a human rather than generated by a machine.	1.0	86.8
Oppose	The insight is wordy	11.2	98.0
	The insight is wrong	1.5	99.5
	The insight lacks detail	0.5	100.0

Procedure Each evaluation session began with a short tutorial presentation that covered an overview of QtoINS, as well as a step-by-step walkthrough demonstrating a typical workflow that covers all the features in QtoINS.

Participants completed one iteration of the analytical process by following the walkthrough. At the end of the tutorial, we introduced four datasets: two from the NLV benchmark dataset [68] repository (cars and movies [9]), one from Kaggle (insurance [10]), and one from the NL4DV Github Repository (Olympic medals [11]). A few sample questions were provided to help participants get started with their data analysis. Participants could choose any dataset. According to our observations, all participants explored at least two datasets.

The tutorial lasted approximately 5–10 minutes. Afterward, participants explored the system at their own pace for 30–60 minutes. During this period, they were encouraged to think aloud and provide verbal feedback. A session coordinator posed follow-up questions as needed to better understand participants' reasoning behind specific interpretations or system feedback. To encourage participants to provide thoughtful open-ended feedback, the remaining portion of the survey was intentionally designed to be concise. It included questions regarding participants' prior experience with data analysis and LLMs (Table 7), Likert-scale ratings for the three main system components

---

[9] https://github.com/nlvcorpus/nlvcorpus.github.io/tree/main/datasets

[10] https://www.kaggle.com/datasets/mirichoi0218/insurance/data

[11] https://github.com/nl4dv/nl4dv/blob/master/examples/assets/data/

Table 5: Supportive and opposing reasons for insights generated by GPT-4o API, together with the percent and accumulations.

	Reason	Pct.(%)	Accum.
Support	The insight is easy to understand	40.1	40.1
	The insight is detailed	10.9	51.0
	The conclusion is useful	6.3	57.3
	The structure of the insight is organized	4.5	61.8
	The description of the chart is well-written	3.2	65.0
Oppose	The conclusion of this chart is simple	14.6	79.6
	The insight is confusing and not clear on its information	7.7	87.3
	The insight is not very detailed	7.6	94.9
	The insight generated is slightly misleading for me	3.2	98.1
	The insight is wordy	1.3	99.4
	The layout of the insight is not good	0.6	100.0

(summarized in Figure 10), and open-ended questions on key system features.

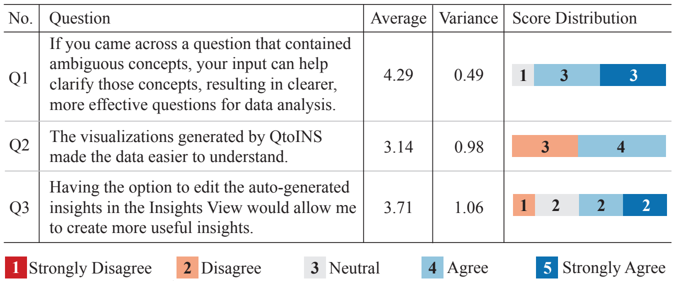
Results CS faculty and staff provided valuable feedback on QtoINS (comments in Appendix C), highlighting both strengths and areas for improvement. Participants appreciated features such as ambiguity resolution, the ability to refine outputs, and charts generation. Their high ratings for Q1 in Figure 10 reflect their improved confidence in clarified questions and verified data. But, they raised concerns about handling extreme values, noted duplication in CoT-generated insights (lower Q3 ratings), and suggested to expand Vega-Lite chart interactions for usability. We implemented three key improvements by incorporating these comments:

- **Revised handling of extreme values:** Instead of returning only a maximum or minimum value, QtoINS now displays the distribution with the extreme value highlighted. This change was implemented by detecting SQL aggregate functions (e.g., `LIMIT 1`, `MAX()`) and intentionally removing them, providing context for comparisons.
- **Reduced overlapping content in insights:** We refined the CoT reasoning prompt to reduce redundancy. ① Observation of the Chart, now focuses only on chart type, axes, and categories/legends, while pattern interpretation is reserved for ② Visual Trend Analysis. When statistical and visual findings align in ④ Integrated Findings, they are summarized in a single concise sentence. Finally, ⑤ Overall Insight Summary, avoids restating earlier steps and instead highlights potential causes, behavioral patterns, or business implications suggested by the chart. These refinements minimize repetition and yield more diverse, higher-quality insights. Later, in an online study with 20 Prolific participants, CoT-generated insights were rated higher than those from GPT-4o (Section 5.3); in the comparative usability study with ChatGPT, ratings for CoT-generated insights also improved.
- **Enhanced chart interactivity:** Vega-Lite charts were extended with additional interactions to support more flexible exploration. Beyond export chart and zooming, we added hovering and tooltips across all nine chart types to facilitate more detailed inspection.

Importantly, many participants raised concerns about how our system compares with ChatGPT, which motivated us to conduct a follow-up comparative study to examine whether QtoINS aligns better with human intent and supports exploratory data analysis.

7 COMPARATIVE USABILITY STUDY

We conducted an online study with 18 non-CS participants from 11 domains (e.g., telecommunications, public health, psychology, finance).



No.	Question	Average	Variance	Score Distribution
Q1	If you came across a question that contained ambiguous concepts, your input can help clarify those concepts, resulting in clearer, more effective questions for data analysis.	4.29	0.49	1 3 3
Q2	The visualizations generated by QtoINS made the data easier to understand.	3.14	0.98	3 4
Q3	Having the option to edit the auto-generated insights in the Insights View would allow me to create more useful insights.	3.71	1.06	1 2 2 2

1 Strongly Disagree 2 Disagree 3 Neutral 4 Agree 5 Strongly Agree

Fig. 10: Ratings for three survey questions, each corresponding to a core component of our system. The table reports the mean score and a breakdown of individual responses on a 5-point Likert scale (1 = strongly disagree, 5 = strongly agree).

They were asked to perform data analysis using both QtoINS and ChatGPT.

Participants All participants were recruited through social media posts, and the study was conducted via online meeting calls (Appendix E Table 10). We recruited 18 participants (12 male, 6 female; ages $21-30$, $\mu = 24.4$, $\sigma = 3$). All reported prior data analysis experience, ranging from < 1 year to > 5 years. None were computer science majors or professional programmers, though all had basic programming knowledge through their undergraduate curriculum. Five (5) participants held High school degrees, 8 bachelor's degrees, 4 master's degrees, and 1 a doctoral degree. Excel (15) and Python (13) were the most widely used tools, followed by Matlab (7), R (5), SQL (3), Tableau (1), and Stata (1). Data analysis frequency varied: 3 reported daily use, 8 weekly, 4 occasional, and 3 monthly. All participants reported some prior experience with LLMs.

Experimental Design We used a within-subject design, for each participant experienced both QtoINS and ChatGPT. To mitigate the learning effect, we adopted a Latin Square to order systems, task types, and datasets (see Table 6). The study examined two main factors: system (QtoINS vs. ChatGPT) and group (using QtoINS first vs. ChatGPT first).

Table 6: Latin square design used in the comparative usability study. Each participant experienced both QtoINS and ChatGPT under a within-subject design. The order of systems, task types (low-level tasks labeled as Task 1 vs. high-level tasks labeled as Task 2), and datasets were counterbalanced using a Latin square to mitigate learning effects.

		QtoINS		ChatGPT		
		Task 1	Task 2	Task 1	Task 2	Participant
Group A	1	Movies	Cars	Medals	Insurance	U1, U5, U17
	2	Cars	Medals	Insurance	Movies	U2, U11
	3	Medals	Insurance	Movies	Cars	U3, U13
	4	Insurance	Movies	Cars	Medals	U4, U15
		ChatGPT		QtoINS		
		Task 1	Task 2	Task 1	Task 2	
Group B	5	Movies	Cars	Medals	Insurance	U6, U10
	6	Cars	Medals	Insurance	Movies	U7, U12
	7	Medals	Insurance	Movies	Cars	U8, U14, U18
	8	Insurance	Movies	Cars	Medals	U9, U16

For the ChatGPT condition, participants used the ChatGPT Data Analysis Plugin to analyze data by uploading files. Participants in Group A used QtoINS first and then ChatGPT, while Group B experienced the reverse order. A short demo (5 minutes) was provided immediately before each system. Then, participants were introduced to the task goals: answering predefined queries for low-level tasks (Task 1 included 24 tasks = 6 tasks x 4 datasets, Appendix D.1), and deriving insights for high-level instructions (Task 2, Appendix D.2). Each task took approximately ($5-10$ minutes). After completing the main experiment ($30-40$ minutes), participants filled out a short survey (5 minutes) and joined a 10-minute discussion to provide comparative feedback on QtoINS and ChatGPT, as well as reflect on notable behaviors (e.g., query clarification, insight editing) observed by the coordinator. The total session duration was approximately 60 minutes.

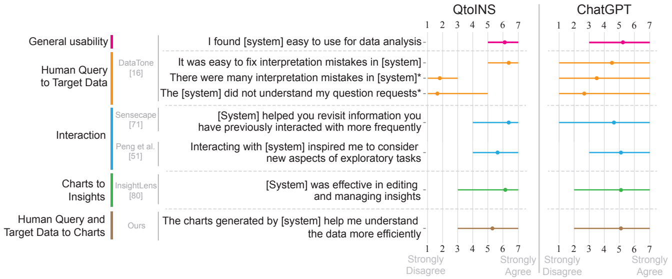
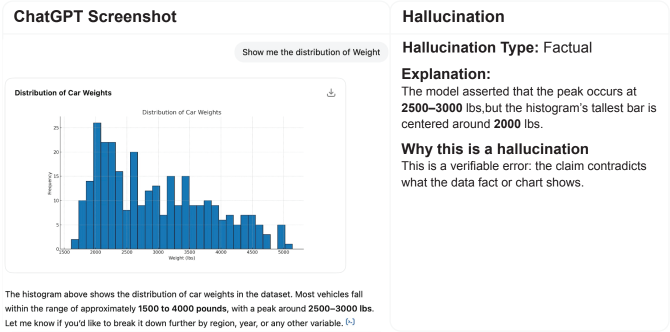


Fig. 11: Subjective ratings on an eight-item questionnaire comparing QtoINS and ChatGPT, measured on a 7-point Likert scale by 18 participants. Bars show mean ratings with min-max intervals. An asterisk indicates negatively worded questions. QtoINS received higher ratings on all positively worded items and lower ratings on negatively worded items, indicating overall better perceived usability for data analysis.

Measurement We collected the following quantitative and qualitative measures:

- **Accuracy:** We measured accuracy in low-level tasks to assess whether participants obtained correct answers in each system. Responses were coded as *Correct* if the answer was complete and accurate, *Incorrect* if entirely wrong, and *Partially Correct* if some values in a column were missed or if the answer was incomplete.
- **Interaction counts per low-level analytical task:** We measured the number of interactions as an indicator of how many times participants refine a query to achieve their analytical goals. For QtoINS, each query was counted as one interaction, while interface actions (e.g., sliders, checkboxes) were excluded. For ChatGPT, each query was counted as one interaction. This metric reflects the efficiency of the interaction process, with fewer counts indicating smoother goal achievement and larger counts suggesting more effort to reach satisfactory results.
- **Subjective rating:** We designed an eight-item questionnaire (listed in Figure 11) by selecting questions from prior instruments on ambiguity management [16], sensemaking [71], exploratory data analysis [51], and insight quality [80], retaining those most relevant to our comparative aims.

Results and Discussion We report results from three perspectives: (1) hallucination issues in data analysis, (2) behavioral changes following increased awareness of these issues, and (3) perceived usability. Detailed participant comments are summarized in Appendix E.



ChatGPT Screenshot

Show me the distribution of Weight

The histogram above shows the distribution of car weights in the dataset. Most vehicles fall within the range of approximately **1500 to 4000 pounds**, with a peak around **2500–3000 lbs**. Let me know if you'd like to break it down further by region, year, or any other variable.

Hallucination

Hallucination Type: Factual

Explanation:
The model asserted that the peak occurs at **2500–3000** lbs,but the histogram's tallest bar is centered around **2000** lbs.

Why this is a hallucination
This is a verifiable error: the claim contradicts what the data fact or chart shows.

Fig. 12: **Factual** hallucination example in ChatGPT.

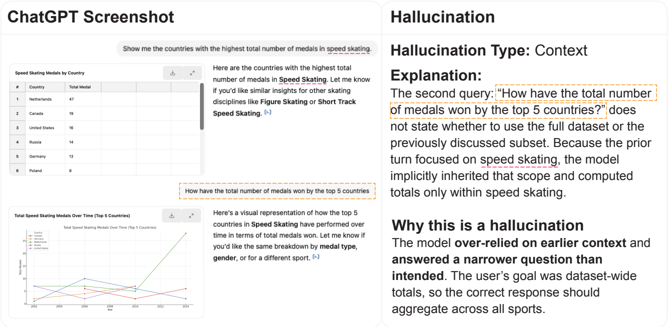
- **Hallucination issues:** We highlight the hallucination issues reported by participants when interacting with ChatGPT; these issues were less frequent but not entirely absent in QtoINS. Our system mitigates hallucination by clarifying ambiguous concepts and values through human-in-the-loop verification, applying a CoT-based approach for insight generation, and structuring components to function independently with inspectable intermediate results. This design increases transparency and reduces the risk of cascading errors from upstream to downstream tasks.

  Based on survey results regarding hallucination in LLMs [58], four types are identified: context, content, semantic, and factual. In our study, no semantic hallucinations were observed with ChatGPT, as prompts were crafted with direct instructions and Python code was generated to analyze the data file. A common example of semantic hallucination is reversing a negative test result into a positive diagnosis.

  Factual hallucination was most commonly observed; for example, in Figure 12, the peak occurs at 2000 lbs, whereas ChatGPT reported a peak around $2500-3000$ lbs. We observed two contrasting participant strategies when handling this error. U4, when faced with a discrepancy between the chart and the textual insight generated by ChatGPT, chose to trust the chart, reasoning that the chart was generated directly from the underlying data and therefore less error-prone. By contrast, U7 was misled by highlighted textual output, accepting the incorrect range of $2500-3000$ lbs as reliable. This case illustrates a "false alarm" effect, where highlighting incorrect insights draws undue attention and increases the likelihood of error. These findings suggest that highlighting functions should be used with caution, as careless inspection may exacerbate rather than mitigate factual hallucinations.

  An example of context hallucination is shown in Figure 13. The query "How have the total number of medals won by the top 5 countries?" was intended to cover the entire dataset. However, because the prior turn focused on speed skating, ChatGPT implicitly narrowed its scope and reported totals only for that sport. This illustrates context hallucination, where the model over-relied on earlier context and produced an incorrect answer. We also observed a unique behavior from U7 (Group A, who used ChatGPT before QtoINS), who habitually added the constraint "in this dataset" to every query. In discussion, U7 explained: "*I frequently use LLM applications like ChatGPT, which often process information beyond the given data, so I developed the habit of adding such constraints.*" This suggests that proficient LLM users can recognize such issues and develop customized prompting strategies to mitigate them.

  Figure 14 shows an example of content hallucination. The model recommended linear regression simply because two numeric variables were present, not because the data supported a linear relationship. The scatterplot displayed an unstructured cloud, so


Speed Skating Medals by Country
Total Speed Skating Medals Over Time (Top 5 Countries)
Total Speed Skating Medals Over Time (Top 5 Countries)
Country
Canada
Germany
Netherlands
Russia
United States
Total Medals
Year
2002
2004
2006
2008
2010
2012
2014
0
5
10
15
20
25

ChatGPT Screenshot

Show me the countries with the highest total number of medals in speed skating.

#	Country	Total Medal
1	Netherlands	47
2	Canada	19
3	United States	16
4	Russia	14
5	Germany	13
6	Poland	8

Here are the countries with the highest total number of medals in Speed Skating. Let me know if you'd like similar insights for other skating disciplines like **Figure Skating** or **Short Track Speed Skating.**

How have the total number of medals won by the top 5 countries

Here's a visual representation of how the top 5 countries in **Speed Skating** have performed over time in terms of total medals won. Let me know if you'd like the same breakdown by **medal type**, **gender**, or for a different sport.

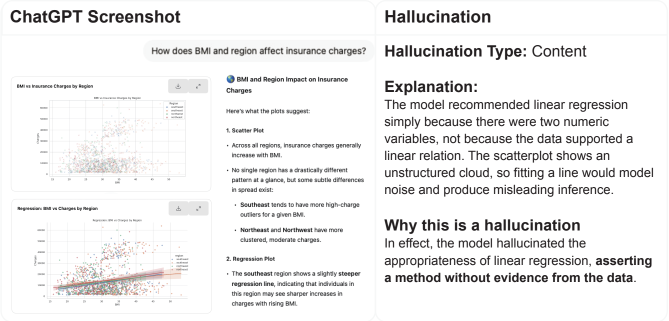
Hallucination

Hallucination Type: Context

Explanation:
The second query: "How have the total number of medals won by the top 5 countries?" does not state whether to use the full dataset or the previously discussed subset. Because the prior turn focused on speed skating, the model implicitly inherited that scope and computed totals only within speed skating.

Why this is a hallucination
The model **over-relied on earlier context** and **answered a narrower question than intended**. The user's goal was dataset-wide totals, so the correct response should aggregate across all sports.

Fig. 13: **Context** hallucination example in ChatGPT.


BMI vs Insurance Charges by Region
BMI vs Insurance Charges by Region
Region
southwest
southeast
northwest
northeast
Charges
BMI
Regression: BMI vs Charges by Region
Regression: BMI vs Charges by Region
region
0
10000
20000
30000
40000
50000
60000
15
20
25
30
35
40
45
50

ChatGPT Screenshot

How does BMI and region affect insurance charges?

🌎 **BMI and Region Impact on Insurance Charges**

Here's what the plots suggest:

1. Scatter Plot

- Across all regions, insurance charges generally increase with BMI.
- No single region has a drastically different pattern at a glance, but some subtle differences in spread exist:
  - **Southeast** tends to have more high-charge outliers for a given BMI.
  - **Northeast** and **Northwest** have more clustered, moderate charges.

2. Regression Plot

- The **southeast** region shows a slightly **steeper regression line**, indicating that individuals in this region may see sharper increases in charges with rising BMI.

Hallucination

Hallucination Type: Content

Explanation:
The model recommended linear regression simply because there were two numeric variables, not because the data supported a linear relation. The scatterplot shows an unstructured cloud, so fitting a line would model noise and produce misleading inference.

Why this is a hallucination
In effect, the model hallucinated the appropriateness of linear regression, **asserting a method without evidence from the data**.

Fig. 14: **Content** hallucination example in ChatGPT.

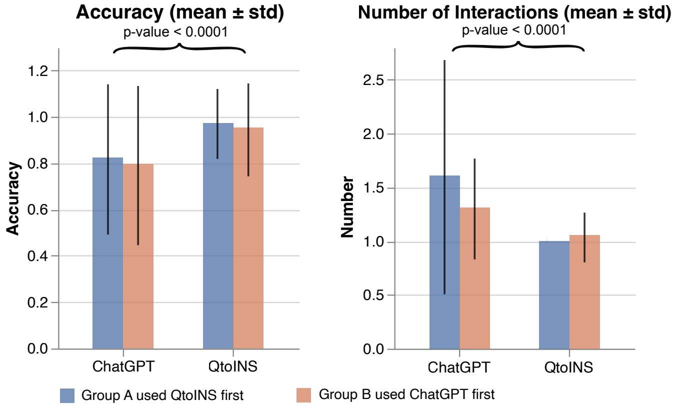


Fig. 15: Average interaction per low-level question and correct answer counts across two systems. Left: Average number of interactions per low-level question, with ChatGPT requiring more refinements, especially after participants experienced QtoINS (Group A). Right: Average correct answer counts, with QtoINS yielding significantly more correct answers compared with ChatGPT across groups.

fitting a line would model noise and lead to misleading inferences. In this case, the model hallucinated the appropriateness of linear regression by asserting a method without evidence from the data. We also observed that ChatGPT can be overly sensitive to perceived problems, proposing additional steps rather than considering the actual data context. Some participants (U1, U9) acknowledged that LLMs are not always reliable but still preferred to use their outputs, reflecting automation bias, the tendency to trust and adopt machine-generated predictions despite potential errors, particularly in low-stakes scenarios [50, 92].

- **Behavioral changes:** For each measurement (interaction count and correct answer count), we examined the two main effects (system, group), as well as their two-way interaction. Across 216 trials (6 questions in Task 1 × 2 systems × 18 participants), a two-way ANOVA in JMP® revealed that the system factor had a significant effect on accuracy ($p < 0.0001$). Table 9 in Appendix D.3 lists the accuracy and number of interactions for all 216 trials. QtoINS achieved a mean accuracy of 0.96 ($\sigma = 0.03$), compared to 0.81 ($\sigma = 0.11$) for ChatGPT. For interaction count, the system factor showed an even stronger effect ($p < 0.0001$), with QtoINS averaging 1.03 interactions ($\sigma = 0.03$) and ChatGPT averaging 1.46 ($\sigma = 0.72$).
  As shown in Figure 15(left), the average number of interactions was significantly higher for ChatGPT, particularly in Group A (who used QtoINS first). After experiencing QtoINS proactive clarification feature, participants became more aware of hallucination risks in ChatGPT and tended to refine their queries more frequently. For example, U7 stated "*After seeing that QtoINS included an active clarification feature, I assumed it would not process information outside the dataset, and therefore stopped adding the constraint 'in this dataset'*". In Figure 15(right), QtoINS achieved a significantly higher number of correct answers regardless of which system was tested first. This improvement reflects the benefit of ambiguity resolution, which was recognized by nearly all participants (16 out of 18).

- **Perceived usability:** In Figure 11, QtoINS received higher average ratings on positively worded items and lower ratings on negatively worded items. These results indicate that QtoINS was perceived more favorably in terms of general usability, ambiguity resolution, multi-view interaction, insight generation, and chart generation for exploratory data analysis.

8 SUMMARY OF KEY RESULTS

We conducted three evaluations. The first two: (i) a component-level evaluation and (ii) a design study guided iterative improvements to both system performance and usability, enabling us to identify and address issues before the comparative user study. We then conducted (iii) a comparative user study (QtoINS vs. ChatGPT) to demonstrate the overall effectiveness of our system for exploratory data analysis.

- Component-level evaluation (Section 5): Does each component work reliably?
  - **Human query to target data:** Our system achieved 98% accuracy, outperforming ChatGPT-4.5 (84%) and NL4DV-LLM (70%).
  - **Chart generation:** The ablation test shows that across 1,080 test cases spanning 7 chart types, our fine-tuned model achieves the best overall accuracy (97.69%) and continues to perform well on popular charts (e.g., bar/pie near or above 99%). The ablation test indicates that combining human query and data feature outperforms fine-tuning with either alone or no fine-tuning; even with a small dataset, 50 samples yielded a 3.6% accuracy gain. The baseline comparison shows that, using 48 chart cases, our system achieved 97.9%, outperforming NL4DV-LLM (83.3%) and chartgpt (72.9%).
  - **Insights generation:** The crowdsourced study shows that the insight generated by our CoT-based approach are better with those generated by the GPT-4o API. CoT-generated insights ($\mu = 4.19$, $\sigma = 0.96$) were rated significantly higher than GPT-4o API-generated insights ($\mu = 3.65$, $\sigma = 1.12$), with a paired-sample t-Test confirming the difference ($p < .0001$).
- Design Study evaluation (Section 6): How can we make QtoINS easier to use?
  - We improve QtoINS by refining extreme-value handling, reducing redundant reasoning in insights, and enhancing chart interactivity,
- Comparative User Study (Section 7): Is QtoINS really easy for casual experts to perform data analysis?
  - **Accuracy and interaction counts for per low-level analytical task:** Overall, QtoINS shows statistically significant improvements over ChatGPT, with fewer interaction counts and higher accuracy.
  - **Subjective rating:** QtoINS was perceived more favorably in terms of general usability, ambiguity resolution, multi-view interaction, insight generation, and chart generation for exploratory data analysis than ChatGPT.

9 DISCUSSION

9.1 End-to-end vs. Compositional Reasoning

Many existing systems [37, 56, 93] perform tasks using an *End-to-End* approach, generating charts or insights in one shot from human queries. This approach is often faster. but lacks transparency, human query validation like QtoINS. Our system breaks down complex data analysis tasks into smaller and manageable steps. This design not only enables human validation, but also facilitates developers to improve individual components independently. If the generated chart is not desirable, for example, due to limited sample size in the fine-tuning dataset, only the second component (② in Figure 2) needs to be upgraded, leaving others untouched.

9.2 Dataset

In practice, datasets are often noisy, inconsistent, or incomplete [29]. Our ambiguity resolution mechanism helps address noisy and inconsistent values. For example, in a medal dataset, the origin column included variations such as "US-1" and "US-2", which our system identified and prompted humans to resolve the ambiguity. QtoINS currently does not handle incomplete data, but future extensions could integrate techniques that handling missing data, such as ReMasker [14]. Upon dataset submission, our system could ask humans to determine whether such algorithms should be applied to augment the data prior to analysis.

Nonetheless, QtoINS was not originally designed as a data-wrangling tool.

9.3 Data Privacy

Prior studies [38, 74, 90] emphasize the ethical and privacy risks of sharing sensitive organizational data with commercial LLMs. In QtoINS, the first two components (human query to target data, charts generation) require only the data header and the first three rows to be sent through GPT-4o API calls, while ambiguity value resolution (scanning of data content) is handled locally.

For chart generation, our system uses Vega-Lite specifications, stripping the data object to produce charts without transmitting the underlying data. If concerns remain about potential privacy leakage through generated charts, privacy-preserving algorithms [9, 65, 89] can be applied to guarantee privacy rules by modifying the data before producing the final charts.

The component that raises the most privacy concern is insight generation, which relies on the target dataset (a subset of the entire data) to support advanced tasks such as outlier detection and trend analysis. Although privacy could be strengthened by generating insights solely from charts, this approach would come at the cost of reduced analytical accuracy. Another complementary solution is to adopt open-source LLMs running locally, thereby avoiding the need to share sensitive data with commercial LLMs.

9.4 Modularization

Similar to commercial tools such as Cursor, Claude Code, and Codex, which allow users to develop Model Context Protocol (MCP) tools for their specific workflows, our components could also potentially be modularized into individual MCP tools. However, our first module, which handles ambiguities, (shown in Fig. 2), cannot be omitted because it is a key component that our system highlights. Once the ambiguities are resolved, module 2 (Charts) and module 3 (Insights) can be considered as individual MCP tools, offering various functionalities and controllable modules, with human-in-the-loop approval required. Additionally, these modules can potentially be integrated within existing ecosystems, as they can not only be standardized for communication, creating a unified way for expert data analysts to utilize them, but also lower the barrier to facilitate integration for those with limited programming scope, such as having expertise only in Python, to plug into module 2 and/or module 3, since the ambiguities have already been resolved in module 1.

9.5 UI Customization

Our system currently offers five views to support its three core modules, and these views can be easily made optional and customizable. For example, people can choose to disable chart views and display only textual insights, or vice versa, depending on their preferences and the task.

Insight View could be easily updated to support more advanced interactions, such as bookmarking and comparative analysis. As people become more familiar with the system autogenerated insights, the interface can be easily adapted to allow them to refine the displayed content and focus on insights that are most relevant and actionable for their specific goals. This level of customization is expected to enhance user engagement and support more efficient, human-centered analytical workflows.

In addition, since participants are more familiar with tools like Excel, they expressed a desire to export the data, chats, and insights to external files. This would allow them to store, share, and further manipulate the results for reporting. They also shared the need for customizable display options, such as dark and light themes. Participants further suggested saving space in Preview Chart View by displaying only the clarified question and allowing charts to be hidden when needed.

9.6 Response Time

The primary source of response latency in our system stems from the ambiguous value checking and reasoning process triggered by invoking the OpenAI Response API for CoT reasoning during insights generation. This introduces a key trade-off in our workflow between response time and reasoning depth. While optimizing for faster responses could enhance human engagement, it must be balanced against the need for thorough analysis and high-quality insight generation, both of which are the core of effective data analysis. Our system may enable people to configure their preferred model based on desired response time and reasoning depth requirements, providing more flexible adaptation to different usage contexts.

9.7 Natural Conversational Clarification vs Rigid Ambiguity Resolution

We designed a structured ambiguity-resolution mechanism that guides casual experts through a step-wise process using carefully designed form-based widgets aligned with a given data schema. By constraining clarification to data headers and values, this rigid UI helps avoid introducing new ambiguities during the resolution process. Compared with a conversational design, where casual experts clarify intent through natural language may require multiple back-and-forth turns (as shown in our comparative study with ChatGPT, which resulted in higher interaction counts), our approach may seem less flexible. However, it provides more affordances for casual experts (rather than displaying top-5 LLM-generated SQL statements or Python code for professional data scientists), and they can easily tell whether their expected values are shown in the interface. Our system does not completely eliminate the possibility of using natural language feedback to clarify the query; people can submit a rephrased question.

9.8 Limitations and Future Work

Participant Expertise Our crowdsourcing evaluation of the Charts to Insights component did not screen participants beyond recruiting US adults (per IRB policy) and did not collect demographic information, such as data analysis proficiency, visualization expertise, or other indicators of data literacy. Therefore, our results reflect how laypeople (general people without guaranteed analytical or visualization expertise) perceive insight quality, and the findings may differ for casual experts, visualization specialists, or professional data analysts.

Ambiguity Coverage We focus on two ambiguity types (i.e., ambiguous concept and ambiguous value) because our system targets single-table exploratory data analysis for casual experts. Within this scope, human intent is typically driven by selection and aggregation and can be expressed using a limited SQL fragment (SELECT/WHERE/-GROUP BY). We do not claim these two ambiguity types are exhaustive. Although we have limited ambiguity coverage, our comparative user study reveals that it can handle 8 low-level task categories [26] and support more comprehensive exploratory data analysis. Beyond our current scope, natural language data queries may involve additional ambiguity forms and more complex analysis needs that our implementation does not yet address, such as multi-table joins, complex temporal data transformations (e.g., moving averages and cumulative sums, as supported in systems like Data Formulator [78] and Data Formulator 2 [77]), and predictive analysis.

Data Schema Our system assumes that the data schema is available because people provide (upload) a table to be analyzed. In this setting, QtoINS can directly derive the schema (e.g., column headers and value types) from the uploaded dataset and use it to validate generated queries and drive ambiguity-resolution widgets. Unlike prior work that must infer or reconcile schemas across multiple external data sources, we do not address schema discovery for unknown, remote, or multi-source datasets; supporting such scenarios (e.g., automatic schema retrieval, dataset selection, and schema matching) is outside the current scope.

Visual Linting Between Charts and Insights. While our system generates both charts and corresponding textual insights, it currently lacks detailed-level visual linting, explicit visual cues linking specific parts of an insight to the corresponding graphical elements in the chart. Effective visual linting could enhance comprehension and promote more integrated analysis. The relative usefulness of charts and textual insights also depends on user intent, whether the goal is to explore data patterns, validate a hypothesis, or communicate findings. For example, the average value shown in Insight View can be interactively linked to a reference line on the y-axis in the chart. Future work could incorporate visual linting mechanisms that cross-reference narrative insights and charts to better support these diverse analytical goals.

Multi-Table Analysis with Foreign Key Joins. The current implementation operates on a single table, which restricts its ability to model more complex data relationships common in real-world databases. Many analytical tasks require the integration of multiple tables connected by foreign keys, for example, joining a transaction table with a customer demographic table. Supporting such relational joins would enable richer, more context-aware reasoning and insights. This would likely involve extending the system's data model and query logic to support multi-table schemas, including join path discovery, relationship-aware data summarization, and efficient query execution across normalized data.

We plan to add a preprocessing module that handles common multi-table integration challenges, such as resolving column naming conflicts across different tables. The preprocessed output would be presented in a structured interface, such as a tree-style repository view organized by table names. This would help users navigate and understand the dataset schema more effectively. Additionally, future iterations could offer automatic translation of the underlying schema into a domain model class diagram, providing an intuitive abstraction layer that supports reasoning and exploration at a conceptual level.

Enhancing Insights Generation. The hallucinations observed in our user studies underscore the need to improve autogenerated insights and motivate several directions for future work. First, to enhance the reasoning capabilities of LLMs and reduce hallucinations in LLM-generated outputs for data insights, future work could explore the integration of knowledge graph reasoning and graph neural networks. Such approaches have shown promise in guiding LLMs to answer complex logical queries more reliably by leveraging structured knowledge sources [32].

Second, inspired by the work of [4], who systematically analyzed the grammatical structure of insights, it has been suggested that insights should be divided into two parts: Domain Knowledge and Analytic Knowledge. In future work, we plan to adopt this approach by explicitly separating generated insights into these two types and refining each individually. This structured separation aims to produce more refined and meaningful interpretations.

10 CONCLUSION

We present QtoINS, a conversational chatbot system designed to support the complete questions-to-insights process. By decomposing the workflow into three modular components (ambiguity resolution, chart generation, and insight derivation), our system improves both the accuracy and transparency of responses. This structured design enables people to iteratively refine their analytical intent, interact with data more effectively, and build confidence in the generated insights. Our evaluation demonstrates that QtoINS not only supports exploratory data analysis through natural language interactions but also helps people produce insights that closely align with human analytical goals.

ACKNOWLEDGMENTS

REFERENCES

[1] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, INFOVIS '05, pp. 111–117. IEEE, Los Alamitos, Oct. 2005. doi: 10.1109/INFVIS.2005.1532136 3, 6

[2] R. Arias-Hernandez, L. T. Kaastra, T. M. Green, and B. Fisher. Pair Analytics: Capturing reasoning processes in collaborative visual analytics. In *Proceedings of the Hawaii International Conference on System Sciences*, HICSS-44, 10 pages. IEEE, Los Alamitos, Jan. 2011. doi: 10.1109/HICSS.2011.339 10

[3] C. Baik, H. V. Jagadish, and Y. Li. Bridging the semantic gap with SQL query logs in natural language interfaces to databases. In *Proceedings of the IEEE International Conference on Data Engineering*, ICDE 2019, pp. 374–385. IEEE, Los Alamitos, Apr. 2019. doi: 10.1109/ICDE.2019.00041 2

[4] L. Battle and A. Ottley. What do we mean when we say "insight"? A formal synthesis of existing theory. *IEEE Transactions on Visualization and Computer Graphics*, 30(9):6075–6088, Sept. 2024. doi: 10.1109/TVCG.2023.3326698 3, 15

[5] A. Bhaskar, T. Tomar, A. Sathe, and S. Sarawagi. Benchmarking and improving Text-to-SQL generation under ambiguity. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 7053–7074. ACL, Kerrville, TX, Dec. 2023. doi: 10.18653/v1/2023.emnlp-main.436 2

[6] P. Bosc and O. Pivert. SQLf: A relational database language for fuzzy querying. *IEEE Transactions on Fuzzy Systems*, 3(1), 1-17 pages, Feb. 1995. doi: 10.1109/91.366566 4

[7] T. Brown, B. Mann, N. Ryder, et al. Language models are few-shot learners. In *Proceedings of the International Conference on Neural Information Processing Systems*, vol. 33 of *NeurIPS 2020*, pp. 1877–1901. Curran Associates, Red Hook, NY, Dec. 2020. 2, 3

[8] N. Chen, Y. Zhang, J. Xu, K. Ren, and Y. Yang. VisEval: A benchmark for data visualization in the era of large language models. *IEEE Transactions on Visualization and Computer Graphics*, 31(1):1301–1311, Jan. 2025. doi: 10.1109/TVCG.2024.3456320 3, 5, 6

[9] A. R. Chowdhury, D. Glukhov, D. Anshumaan, P. Chalasani, N. Papernot, S. Jha, and M. Bellare. Prεεmpt: Sanitizing sensitive prompts for LLMs. *arXiv preprint arXiv.2504.05147*, 34 pages, Aug. 2025. doi: 10.48550/arXiv.2504.05147 14

[10] D. Deng, A. Wu, H. Qu, and Y. Wu. DashBot: Insight-driven dashboard generation based on deep reinforcement learning. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):690–700, Jan. 2023. doi: 10.1109/TVCG.2022.3209468 3

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, NAACL 2019, pp. 4171–4186. ACL, Kerrville, TX, June 2019. doi: 10.18653/v1/N19-1423 2, 5

[12] V. Dibia. LIDA: A tool for automatic generation of grammar-agnostic visualizations and infographics using large language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, ACL 2023, pp. 113–126. ACL, July 2023. doi: 10.18653/v1/2023.acl-demo.11 2, 3

[13] V. Dibia and Ç. Demiralp. Data2Vis: Automatic generation of data visualizations using sequence-to-sequence recurrent neural networks. *IEEE Computer Graphics and Applications*, 39(5):33–46, Sept. 2019. doi: 10.1109/MCG.2019.2924636 3

[14] T. Du, L. Melis, and T. Wang. ReMasker: Imputing tabular data with masked autoencoding. In *Proceedings of the International Conference on Representation Learning*, ICLR 2024, 23 pages. Curran Associates, Red Hook, NY, May 2024. 13

[15] A. Floratou, F. Psallidas, F. Zhao, S. Deep, G. Hagleither, et al. NL2SQL is a solved problem... Not! In *Proceedings of the Annual Conference on Innovative Data Systems Research*, CIDR 2024, 8 pages, Jan. 2024. 2

[16] T. Gao, M. Dontcheva, E. Adar, Z. Liu, and K. G. Karahalios. DataTone: Managing ambiguity in natural language interfaces for data visualization. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '15, p. 489âĂŞ500. ACM, New York, Nov. 2015. doi: 10.1145/2807442.2807478 2, 3, 12

[17] S. Gathani, A. Crisan, V. Setlur, and A. Srinivasan. GROOT: A system for editing and configuring automated data insights. In *Proceedings of the IEEE Visualization and Visual Analytics*, VIS 2024, pp. 36–40. IEEE, Los Alamitos, Oct. 2024. doi: 10.1109/VIS55277.2024.00015 3

[18] J. Guo, Z. Zhan, Y. Gao, Y. Xiao, J.-G. Lou, T. Liu, and D. Zhang. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 4524–4535. ACL, Florence, Italy, July 2019. doi: 10.18653/v1/P19-1444 2

[19] J. Hong, C. Seto, A. Fan, and R. Maciejewski. Do LLMs have visualization literacy? an evaluation on modified visualizations to test generalization in data interpretation. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–13, Early Access. Accepted on Jan. 29, 2025. doi: 10.1109/TVCG.2025.3536358 3

[20] K. Hu, M. A. Bakker, S. Li, T. Kraska, and C. Hidalgo. VizML: A machine learning approach to visualization recommendation. In *Proceedings of*

the ACM Conference on Human Factors in Computing Systems, CHI '19, article no. 128, 12 pages. ACM, New York, May 2019. doi: 10.1145/3290605.3300358 3, 5, 6

[21] L. Jing, Z. Huang, X. Wang, W. Yao, W. Yu, K. Ma, H. Zhang, X. Du, and D. Yu. DSBench: How far are data science agents from becoming data science experts?, Apr. 2025. doi: 10.48550/arXiv.2409.07703 2

[22] B. Ju and J. B. Stewart. Empowering users with ChatGPT and similar large language models (llms): Everyday information needs, uses, and gratification. *Proceedings of the Association for Information Science and Technology*, 61(1):172âĂŞ182, Oct. 2024. doi: 10.1002/pra2.1018 1

[23] S. Kandel, A. Paepcke, J. M. Hellerstein, and J. Heer. Enterprise data analysis and visualization: An interview study. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2917–2926, Dec. 2012. doi: 10.1109/TVCG.2012.219 2

[24] Y.-a. Kang, C. GÃűrg, and J. Stasko. How can visual analytics assist investigative analysis? design implications from an evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 17(5):570–583, May 2011. doi: 10.1109/TVCG.2010.84 7

[25] Y. Kim and J. Heer. Assessing effects of task and data distribution on the effectiveness of visual encodings. *Computer Graphics Forum*, 37(3):157–167, July 2018. doi: 10.1111/cgf.13409 5

[26] S. Lee, S.-H. Kim, and B. C. Kwon. Vlat: Development of a visualization literacy assessment test. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):551–560, Jan. 2017. doi: 10.1109/TVCG.2016.2598920 14

[27] F. Lei, J. Chen, Y. Ye, R. Cao, D. Shin, H. Su, Z. Suo, H. Gao, W. Hu, P. Yin, V. Zhong, C. Xiong, R. Sun, Q. Liu, S. Wang, and T. Yu. Spider 2.0: Evaluating language models on real-world enterprise Text-to-SQL workflows. *arXiv preprint arXiv.2411.07763*, 45 pages, Mar. 2025. doi: 10.48550/arXiv.2411.07763 2

[28] F. Lei, Y. Ma, A. S. Fotheringham, E. A. Mack, Z. Li, M. Sachdeva, S. Bardin, and R. Maciejewski. GeoExplainer: A visual analytics framework for spatial modeling contextualization and report generation. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1391–1401, Jan. 2024. doi: 10.1109/TVCG.2023.3327359 3

[29] A. Lew, M. Agrawal, D. Sontag, and V. Mansinghka. PClean: Bayesian data cleaning at scale with domain-specific probabilistic programming. In *Proceedings of The International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, pp. 1927–1935. PMLR, Cambridge, MA, Apr. 2021. 13

[30] X. Li, J. Zhou, W. Chen, D. Xu, T. Xu, and E. Chen. Visualization recommendation with prompt-based reprogramming of large language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2024, pp. 13250–13262. ACL, Aug. 2024. doi: 10.18653/v1/2024.acl-long.716 3

[31] Y. Lin, H. Li, L. Yang, A. Wu, and H. Qu. Inksight: Leveraging sketch interaction for documenting chart findings in computational notebooks. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):944âĂŞ954, Jan. 2024. doi: 10.1109/TVCG.2023.3327170 2

[32] L. Liu, Z. Wang, R. Qiu, Y. Ban, E. Chan, Y. Song, J. He, and H. Tong. Logic query of thoughts: Guiding large language models to answer complex logic queries with knowledge graphs. *arXiv preprint arXiv:2404.04264*, 15 pages, Nov. 2024. doi: 10.48550/arXiv.2404.04264 15

[33] A. Lundgard and A. Satyanarayan. Accessible visualization via natural language descriptions: A four-level model of semantic content. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1073–1083, Jan. 2022. doi: 10.1109/TVCG.2021.3114770 6

[34] T. Luo, C. Huang, L. Shen, B. Li, S. Shen, W. Zeng, N. Tang, and Y. Luo. nvBench 2.0: A benchmark for natural language to visualization under ambiguity. *arXiv preprint arXiv.2503.12880*, 14 pages, Mar. 2025. doi: 10.48550/arXiv.2503.12880 3

[35] Y. Luo, X. Qin, N. Tang, and G. Li. DeepEye: Towards automatic data visualization. In *Proceedings of the IEEE International Conference on Data Engineering*, ICDE 2018, pp. 101–112. IEEE, Los Alamitos, Apr. 2018. doi: 10.1109/ICDE.2018.00019 3

[36] Y. Luo, N. Tang, G. Li, C. Chai, W. Li, and X. Qin. Synthesizing natural language to visualization (NL2VIS) benchmarks from NL2SQL benchmarks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, SIGMOD '21, p. 1235âĂŞ1247. ACM, New York, June 2021. doi: 10.1145/3448016.3457261 3

[37] Y. Luo, N. Tang, G. Li, J. Tang, C. Chai, and X. Qin. Natural language to visualization by neural machine translation. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):217–226, Jan. 2022. doi: 10.1109/TVCG.2021.3114848 3, 6, 13

[38] R. Ma, C. Maidhof, J. C. Carrillo, J. Lindqvist, and J. Such. Privacy perceptions of custom GPTs by users and creators. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, CHI '25, article no. 237, 18 pages. ACM, New York, Apr. 2025. doi: 10.1145/3706598.3713540 14

[39] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5(2):110âĂŞ141, Apr. 1986. doi: 10.1145/22949.22950 3

[40] J. Mackinlay, P. Hanrahan, and C. Stolte. Show Me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, Nov. 2007. doi: 10.1109/TVCG.2007.70594 3

[41] K. Madhavan, N. Elmqvist, M. Vorvoreanu, X. Chen, Y. Wong, H. Xian, Z. Dong, and A. Johri. DIA2: Web-based cyberinfrastructure for visual analysis of funding portfolios. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1823–1832, Dec. 2014. doi: 10.1109/TVCG.2014.2346747 2, 7

[42] R. Mitra, A. Narechania, A. Endert, and J. Stasko. Facilitating conversational interaction in natural language interfaces for visualization. In *2022 IEEE Visualization and Visual Analytics (VIS)*, pp. 6–10, 2022. doi: 10.1109/VIS54862.2022.00010 2

[43] D. Moritz, C. Wang, G. L. Nelson, H. Lin, A. M. Smith, B. Howe, and J. Heer. Formalizing visualization design knowledge as constraints: Actionable and extensible models in Draco. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):438–448, Jan. 2019. doi: 10.1109/TVCG.2018.2865240 3

[44] T. Munzner. *Visualization Analysis and Design*, chap. 3.6, pp. 57–58. CRC Press, Boca Raton, FL, 2014. 4

[45] A. Narechania, A. Srinivasan, and J. Stasko. NL4DV: A toolkit for generating analytic specifications for data visualization from natural language queries. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):369–379, Feb. 2021. doi: 10.1109/TVCG.2020.3030378 2, 3, 6

[46] OpenAI. Openai model comparison. `https://platform.openai.com/docs/models/compare`, 2024. Accessed: 2025-01-12. 8, 27

[47] OpenAI, J. Achiam, S. Adler, S. Agarwal, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774v6*, 23 pages, Mar. 2024. doi: 10.48550/arXiv.2303.08774 2, 3

[48] E. Oral, R. Chawla, M. Wijkstra, N. Mahyar, and E. Dimara. From information to choice: A critical inquiry into visualization tools for decision making. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):359âĂŞ369, Jan. 2024. doi: 10.1109/TVCG.2023.3326593 3

[49] L. Pan, M. Saxon, W. Xu, D. Nathani, X. Wang, and W. Y. Wang. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506, July 2024. doi: 10.1162/tacl_a_00660 2

[50] R. Parasuraman and D. H. Manzey. Complacency and bias in human use of automation: An attentional integration. *Human Factors*, 52(3):381–410, Oct 2010. doi: 10.1177/0018720810376055 13

[51] Y. Peng, X. Qin, Z. Zhang, J. Zhang, Q. Lin, X. Yang, D. Zhang, S. Rajmohan, and Q. Zhang. Navigating the unknown: A chat-based collaborative interface for personalized exploratory tasks. In *Proceedings of the International Conference on Intelligent User Interfaces*, IUI '25, p. 1048âĂŞ1063. ACM, New York, Mar. 2025. doi: 10.1145/3708359.3712093 12

[52] L. Podo, B. Prenkaj, and P. Velardi. Agnostic visual recommendation systems: Open challenges and future directions. *IEEE Transactions on Visualization and Computer Graphics*, 31(3):1902–1917, Mar. 2025. doi: 10.1109/TVCG.2024.3374571 2

[53] L. Qiu, J. Li, C. Su, and L. Chen. Interactive text-to-SQL via expected information gain for disambiguation. *arXiv preprint arXiv.2507.06467*, 13 pages, July 2025. doi: 10.48550/arXiv.2507.06467 2

[54] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140), 67 pages, June 2020. 2

[55] D. Sacha, A. Stoffel, F. Stoffel, B. C. Kwon, G. Ellis, and D. A. Keim. Knowledge generation model for visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1604–1613, Nov. 2014. doi: 10.1109/TVCG.2014.2346481 2, 3

[56] S. Sah, R. Mitra, A. Narechania, A. Endert, J. Stasko, and W. Dou. Generating analytic specifications for data visualization from natural language

queries using large language models. In *IEEE VIS Workshop on Natural Language Interfaces for Visualization and Visual Analytics*, NLVIZ 2024, 6 pages. IEEE, Los Alamitos, Oct. 2024. doi: 10.48550/arXiv.2408.13391 2, 6, 9, 13

[57] D. Saha, A. Floratou, K. Sankaranarayanan, U. F. Minhas, A. R. Mittal, and F. Özcan. ATHENA: An ontology-driven system for natural language querying over relational data stores. *Proceedings of the VLDB Endowment*, 9(12):1209âĂŞ1220, Aug. 2016. doi: 10.14778/2994509.2994536 2

[58] P. Sahoo, P. Meharia, A. Ghosh, S. Saha, V. Jain, and A. Chadha. A comprehensive survey of hallucination in large language, image, video and audio foundation models. In *Findings of the Association for Computational Linguistics*, EMNLP 2024, pp. 11709–11724. ACL, Kerrville, TX, Nov. 2024. doi: 10.18653/v1/2024.findings-emnlp.685 12

[59] B. Saket, A. Endert, and Ã. Demiralp. Task-based effectiveness of basic visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 25(7):2505–2512, May 2019. doi: 10.1109/TVCG.2018.2829750 3

[60] I. Saparina and M. Lapata. AMBROSIA: A benchmark for parsing ambiguous questions into database queries. In *Proceedings of the International Conference on Neural Information Processing Systems*, vol. 37 of *NeurIPS 2024*, 29 pages, pp. 90600–90628. Curran Associates, Red Hook, NY, Dec. 2024. 2

[61] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-Lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, Jan. 2017. doi: 10.1109/TVCG.2016.2599030 5

[62] H.-J. Schulz, T. Nocke, M. Heitzler, and H. Schumann. A design space of visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2366–2375, Oct. 2013. doi: 10.1109/TVCG.2013.120 3

[63] V. Setlur, S. E. Battersby, M. Tory, R. Gossweiler, and A. X. Chang. Eviza: A natural language interface for visual analysis. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '16, p. 365âĂŞ377. ACM, New York, Oct. 2016. doi: 10.1145/2984511.2984588 2, 3

[64] V. Setlur, M. Tory, and A. Djalali. Inferencing underspecified natural language utterances in visual analysis. In *Proceedings of the International Conference on Intelligent User Interfaces*, IUI '19, p. 40âĂŞ51. ACM, New York, Mar. 2019. doi: 10.1145/3301275.3302270 6

[65] Z. Shen, Z. Xi, Y. He, W. Tong, J. Hua, and S. Zhong. The fire thief is also the keeper: Balancing usability and privacy in prompts. *arXiv preprint arXiv.2406.14318*, 16 pages, June 2024. doi: 10.48550/arXiv.2406.14318 14

[66] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu. MPNet: Masked and permuted pre-training for language understanding. In *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS 2020, article no. 1414, 11 pages. Curran Associates, Red Hook, NY, Dec. 2020. 4, 28

[67] A. Srinivasan, S. M. Drucker, A. Endert, and J. Stasko. Augmenting visualizations with interactive data facts to facilitate interpretation and communication. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):672–681, Jan. 2019. doi: 10.1109/TVCG.2018.2865145 3

[68] A. Srinivasan, N. Nyapathy, B. Lee, S. M. Drucker, and J. Stasko. Collecting and characterizing natural language utterances for specifying data visualizations. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, CHI '21, article no. 464, 10 pages. ACM, New York, May 2021. doi: 10.1145/3411764.3445400 2, 8, 9, 10

[69] A. Srinivasan, V. Setlur, and A. Satyanarayan. Pluto: Authoring semantically aligned text and charts for data-driven communication. In *Proceedings of the International Conference on Intelligent User Interfaces*, IUI '25, p. 1123âĂŞ1140. ACM, New York, Mar. 2025. doi: 10.1145/3708359.3712122 2

[70] C. Stokes, V. Setlur, B. Cogley, A. Satyanarayan, and M. A. Hearst. Striking a Balance: Reader takeaways and preferences when integrating text and charts. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1233–1243, Jan. 2023. doi: 10.1109/TVCG.2022.3209383 3, 8

[71] S. Suh, B. Min, S. Palani, and H. Xia. Sensecape: Enabling multilevel exploration and sensemaking with large language models. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '23, 18 pages. ACM, New York, Oct. 2023. doi: 10.1145/3586183.3606756 12

[72] Y. Tian, W. Cui, D. Deng, X. Yi, Y. Yang, H. Zhang, and Y. Wu. ChartGPT: Leveraging LLMs to generate charts from abstract natural language. *IEEE Transactions on Visualization and Computer Graphics*, 31(3):1731–1745, Mar. 2025. doi: 10.1109/TVCG.2024.3368621 2, 3, 6

[73] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. RoziÃĺre, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv.2302.13971*, 16 pages, Feb. 2023. doi: 10.48550/arXiv.2302.13971 3

[74] K. B. Wagman, M. T. Dearing, and M. Chetty. Generative AI uses and risks for knowledge workers in a science organization. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, CHI '25, article no. 1199, 17 pages. ACM, New York, Apr. 2025. doi: 10.1145/3706598.3713827 14

[75] B. Wang, Y. Gao, Z. Li, and J.-G. Lou. Know what I don't know: Handling ambiguous and unknown questions for text-to-SQL. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 5701–5714. ACL, Kerrville, TX, July 2023. doi: 10.18653/v1/2023.findings-acl.352 2

[76] B. Wang, R. Shin, X. Liu, O. Polozov, and M. Richardson. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 7567–7578. ACL, Kerrville, TX, July 2020. 2

[77] C. Wang, B. Lee, S. Drucker, D. Marshall, and J. Gao. Data Formulator 2: Iterative creation of data visualizations, with AI transforming data along the way. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, CHI '25, 17 pages. ACM, New York, Apr. 2025. doi: 10.1145/3706598.3713296 2, 3, 8, 14

[78] C. Wang, J. Thompson, and B. Lee. Data Formulator: AI-powered concept-driven visualization authoring. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1128–1138, Jan. 2024. doi: 10.1109/TVCG.2023.3326585 2, 3, 8, 14

[79] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the International Conference on Neural Information Processing Systems*, NeurIPS 2022, article no. 1800, 14 pages. Curran Associates, Red Hook, NY, Nov. 2022. 3

[80] L. Weng, X. Wang, J. Lu, Y. Feng, Y. Liu, H. Feng, D. Huang, and W. Chen. InsightLens: Augmenting LLM-powered data analysis with interactive insight management and navigation. *IEEE Transactions on Visualization and Computer Graphics*, 31(6):3719–3732, 2025. doi: 10.1109/TVCG.2025.3567131 3, 12

[81] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):649–658, Jan. 2016. doi: 10.1109/TVCG.2015.2467191 3

[82] E. Wu, K. Wu, and J. Zou. FineTuneBench: How well do commercial fine-tuning APIs infuse knowledge into LLMs? *arXiv preprint arXiv.2411.05059*, 14 pages, Nov. 2024. doi: 10.48550/arXiv.2411.05059 6

[83] Y. Wu, M. S. Hee, Z. Hu, and R. K.-W. Lee. Longgenbench: Benchmarking long-form generation in long context LLMs. *arXiv preprint arXiv.2409.02076*, 22 pages, Jan. 2025. doi: 10.48550/arXiv.2409.02076 5

[84] N. Yaghmazadeh, Y. Wang, I. Dillig, and T. Dillig. SQLizer: Query synthesis from natural language. *Proc. ACM Program. Lang.*, article no. 63, 26 pages, Oct. 2017. doi: 10.1145/3133887 2

[85] B. Yu and C. T. Silva. FlowSense: A natural language interface for visual data exploration within a dataflow system. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1–11, Jan. 2020. doi: 10.1109/TVCG.2019.2934668 3

[86] T. Yu, Z. Li, Z. Zhang, R. Zhang, and D. Radev. TypeSQL: Knowledge-based type-aware neural text-to-SQL generation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, NAACL 2018, pp. 588–594. ACL, Kerrville, TX, June 2018. doi: 10.18653/v1/N18-2093 2

[87] T. Yu, M. Yasunaga, K. Yang, R. Zhang, D. Wang, Z. Li, and D. Radev. SyntaxSQLNet: Syntax tree networks for complex and cross-domain text-to-SQL task. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1653–1663. ACL, Kerrville, TX, Oct. 2018. doi: 10.18653/v1/D18-1193 2

[88] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP 2018, pp. 3911–3921. ACL,

Kerrville, TX, Oct. 2018. doi: 10.18653/v1/D18-1425 2, 8

[89] J. Zhan, W. Zhang, Z. Zhang, H. Xue, Y. Zhang, and Y. Wu. Portcullis: A scalable and verifiable privacy gateway for third-party LLM inference. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(1):1022–1030, Apr. 2025. doi: 10.1609/aaai.v39i1.32088 14

[90] Z. Zhang, M. Jia, H.-P. H. Lee, B. Yao, S. Das, A. Lerner, D. Wang, and T. Li. "it's a fair game", or is it? Examining how users navigate disclosure risks and benefits when using LLM-based conversational agents. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, CHI '25, article no. 156, 26 pages. ACM, New York, Apr. 2025. doi: 10.1145/3613904.3642385 14

[91] F. Zhao, S. Deep, F. Psallidas, A. Floratou, D. Agrawal, and A. E. Abbadi. Sphinteract: Resolving ambiguities in NL2SQL through user interaction. *Proceedings of the VLDB Endowment*, 18(4):1145âĂŞ1158, May 2025. doi: 10.14778/3717755.3717772 2

[92] J. Zhao, Y. Wang, M. V. Mancenido, E. K. Chiou, and R. Maciejewski. Evaluating the impact of uncertainty visualization on model reliance. *IEEE Transactions on Visualization and Computer Graphics*, 30(7):4093–4107, July 2024. doi: 10.1109/TVCG.2023.3251950 13

[93] Y. Zhao, J. Wang, L. Xiang, X. Zhang, Z. Guo, C. Turkay, Y. Zhang, and S. Chen. LightVA: Lightweight visual analytics with LLM agent-based task planning and execution. *IEEE Transactions on Visualization and Computer Graphics*, 31(9):6162–6177, Sept. 2025. Accepted on Nov. 14, 2024. doi: 10.1109/TVCG.2024.3496112 2, 3, 13

[94] Y. Zhao, Y. Zhang, Y. Zhang, X. Zhao, J. Wang, Z. Shao, C. Turkay, and S. Chen. LEVA: Using large language models to enhance visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 31(3):1830–1847, Mar. 2024. doi: 10.1109/TVCG.2024.3368060 3

[95] V. Zhong, C. Xiong, and R. Socher. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103, 12 pages, 2017. doi: 10.48550/arXiv.1709.00103 2

A PROMPT EXAMPLES GUIDING LLMS

This appendix provides details of guiding prompts in Section 3 of the full paper, including Human Query to Target Data, Human Query and Target Data to Charts, and Charts to Insights.

A.1 Human Query to Target Data

We provide examples of prompts when using LLMs in our first component: Human Query to Target Data.

A.1.1 Overview

We use a two-step interactive procedure to synthesize validated SQL statements with human intent by leveraging LLMs. Step 1 is resolving ambiguous concepts, and Step 2 is resolving ambiguous values. Sections A.1.2–A.1.4 are in Step 1, and Section A.1.5 is in Step 2.

A.1.2 Wrapped SQL Statements Generation

Table A.1.2 shows the prompt to produce SQL statements with wrapped ambiguous concepts.

Table A.1.2: Producing SQL Statements with Wrapped Ambiguous Concepts (System Prompt)

Your task is to generate an `SQL` statement strictly adhering to DuckDB syntax based on the user's question.
Whenever a concept in the user's request is ambiguous, you must encapsulate it directly within curly braces { } without attempting to infer or clarify the user's intent.
STRICT POSITION RULES FOR { }
The "{ }" placeholder is ONLY allowed in the following three cases but could appear multiple times:

1. In the `SELECT` clause, replacing the name of a non-aggregated column or inside aggregate function (e.g., SUM, MAX, MIN, COUNT,AVG) arguments (e.g., SUM(value) is correct).
2. A placeholder in the `WHERE` clause may be used only when replacing a literal value that is explicitly paired with a specific numeric column. It must appear on the right-hand side of the comparison, and the comparison operator must be one of >, >=, <, <=; the '=' operator is not allowed.
3. A placeholder in the `limit` clause and in the `GROUP BY` clause.

- Example

- Table Name: "olympic_medals"
- Columns: "Name", "Sex", "Age", "Country", "Sport", "Year", "Bronze Medal", "Gold Medal", "Silver Medal", "Total Medal"
- User's question: "Compare the medals for football by country."
- Generated SQL:

```
SELECT Country, {medal} FROM olympic_medals WHERE Sport = 'Football';
```

- Reason: "medal" is ambiguous, medal can be Bronze Medal, Gold Medal, Silver Medal, Total Medal

- Example

- Table Name: "olympic_medals"
- Columns: "Name", "Sex", "Age", "Country", "Sport", "Year", "Bronze Medal", "Gold Medal", "Silver Medal", "Total Medal"
- User's question: "Identify the top 3 older sports with the highest number of athletes after 2016."
- Generated SQL:

```
SELECT Sport, COUNT(*) AS AthleteCount
FROM olympic_medals
WHERE Year > 2016 AND Age > {older}
GROUP BY Sport
ORDER BY AthleteCount DESC
LIMIT 3;
```

- Reason: "older" is ambiguous, older can be 30, 40, 50 or other age

Formatting Instructions

- Encapsulate any ambiguous or unclear terms directly within curly braces { }.
- Do not include any explanatory text, comments, or additional context in the final SQL output.
- If a user's request has some string match, you need to confidently guess simple string mappings based on clear user instructions.
-

Table A.1.3: Producing SQL Statements with Wrapped Ambiguous Concepts (User Prompt)

```
My table name is: [table_name]. The columns in my
table are: [column_name_list].
My first three rows of data are: [table_value].
My question is: [human_query]
```

A.1.3 Identify Column Names

We provide a guiding prompt for the `Identify Column Names` task in Figure 3 to search all possible column names.

Table A.1.4: Identify Column Names (System Prompt)

You are now a knowledgeable expert. I will provide a table and its columns; please use this information to answer my questions. In my case, some parts of my SQL are different. The fuzzy concepts in my SQL are enclosed in . Vague concepts now fall into in this scenarios: When the vague concept in the query is unclear due to multiple possible matches with the table columns and there are at least two possible columns that could match, you must return a Python list containing all potential column names the vague concept might correspond to. Only return this list, without additional information or explanation.

Example for Scenario 1

- Table Name: movies
- Columns: title, director, genre, avg rating, max rating, min rating
- Question: “show me the relationship between rating and title”
- SQL:

```
SELECT {rating}, title FROM movies;
```

- You must return:

```
['avg rating' , 'max rating' , 'min rating' ]
```

Table A.1.5: Identify Column Names (User Prompt)

```
My table name is: [table_name]. The columns in my
table are: [column_name_list].
My question is: [human_query]. My sql is: [sql].
In my case, some parts of my SQL are different.
The fuzzy concepts in my SQL are enclosed in {}.
In this task, you only need to address only one
vague concept: [ambiguous_concept].
My first three rows of data are: [table_value].
```

A.1.4 Validate Dynamic Concepts Defined by Users On-the-fly

We provide a guiding prompt for validating inputs during dynamic concept creation based on human input.

Table A.1.6: Validate Dynamic Concepts Defined by Users On-the-fly (System Prompt)

You are an expert in SQL and creative problem-solving. We have an SQL syntax where ambiguous concepts are enclosed in curly braces `{}`. You will receive:
âĂć A table schema.
âĂć An SQL query containing an ambiguous concept (enclosed in `{}`).
âĂć A proposed textual solution to clarify the ambiguity.

Your task is to categorize whether the provided textual solution logically and completely resolves the ambiguity using ONLY the information available from the given table schema.

Important rules:
- The provided solution must directly and exactly replace the ambiguous concept in the SQL query.
- The replacement must match exactly in spelling, spacing, punctuation, underscores, and capitalization as shown in the schema and original SQL.
- Even the smallest mismatch (missing/extra underscores, wrong case, wrong spacing, punctuation differences, etc.) makes the solution incorrect.

Categorize the resolution into exactly one of these two levels:
1. “fully_resolvable” : The textual solution entirely resolves the ambiguity by clearly using columns and expressions provided explicitly in the table schema. The solution, when directly replacing the ambiguous concept, produces a ready-to-execute SQL query.
2. “completely_unresolvable” : The textual solution references concepts or data not included in or calculable from the given table schema, or does not exactly match the column names or expressions required by the schema.

Return ONLY in this JSON format without any additional commentary or explanation: `{“level” : “” ,}`

Example 1 (fully_resolvable)

- Table Name: “product_sales”
- Columns: “product_id” , “product_name” , “category” , “price” , “units_sold” , “revenue”
- User’s question: “Compare the medals for football by country.”
- SQL:

```
SELECT product_name , {best_selling} as top_products
FROM product_sales
ORDER BY {best_selling} DESC LIMIT 5;
```

- Solution: “price * units_sold”
- Expected JSON response:

```
{'level' : 'fully_resolvable' ,}
```

Example 2 (completely_unresolvable)

- Table Name: “store_inventory”
- Columns: “store_id” , “product_id” , “quantity” , “price”
- Solution: “The most valuable stores are the ones located in high-income zip codes.”
- SQL:
  ```
  SELECT store_id, {most_valuable} FROM store_inventory
  GROUP BY store_id ORDER BY {most_valuable} DESC
  ```
- Solution: The most valuable stores are the ones located in high-income zip codes.
- Expected JSON response:
  ```
  {'level' : 'completely_unresolvable',}
  ```

Example 3 (completely_unresolvable)

- Table Name: “orders”
- Columns: “order_id”, “customer_id”, “order_date”, “total_amount”, “discount_rate”
- Solution: “total amount * (1 - discount rate)”
- SQL:
  ```
  SELECT customer_id, {final_price} as discounte_total FROM orders;
  ```
- Solution: The most valuable stores are the ones located in high-income zip codes.
- Expected JSON response:
- Reason: Column names in the schema are “total_amount” and “discount_rate”. The provided solution uses spaces instead of underscores, so it does not exactly match the schema column names.
  ```
  {'level': 'completely_unresolvable',}
  ```

Table A.1.7: Validate Dynamic Concepts Defined by Users On-the-fly (User Prompt)

```
My table name is: [table_name]. The columns in my
table are: [column_name_list]. My question
is: [human_query]. My sql is: [sql]. In my case,
some parts of my SQL are different.
The fuzzy concepts in my SQL are enclosed in {}.
In this task, The fuzzy concept you need to solve
is: [ambiguous_concept]. The solution is:
[solution]. My first three rows of data are:
[table_value].
```

A.1.5 Resolve Ambiguous Values

We provide two guiding prompts to resolve ambiguous values. If string matching in a `WHERE` clause uses “=”, the system applies Resolve Ambiguous Values 1; if it uses “LIKE”, the system applies Resolve Ambiguous Values 2.

Table A.1.8: Resolve Ambiguous Values 1 (System Prompt)

Task Description

You are tasked with analyzing a user’s query regarding an SQL statement and determining whether a specific string used within an SQL statement’s condition is vague or ambiguous. The user is unaware of the actual content within the database, so your judgment should be based on semantic similarity and reasonable alternatives that might match the user’s intent.

Instructions

Based on the user’s query, table structure, and the provided column, analyze the top strings that are semantically most similar to the target string. If you determine that the string the user provided is ambiguous or fuzzy - meaning that there are other strings in the column that could reasonably fulfill the user’s intent - set the flag to **true** and provide those alternative strings in the `Fuzzy_List`.

Consider broadly synonymous or contextually similar terms, especially when dealing with categorical, geographic, or genre-related queries. The alternatives you choose should reasonably capture the user’s original intent, even if they are not identical to the user’s input. If no suitable alternatives exist that align with the intended meaning, set the flag to **false** without an additional list.

Example 1

Query: Now I have a table named `books` and this table has these columns: Rank, book title, book price, rating, author, year of publication, genre, url. My question is: I want to know which books are in the `'Fiction'` genre. Can you show me the book title, author, and URL?

SQL:

```
SELECT 'book title', author, url
FROM books
WHERE genre = 'Fiction';
```

Similar Strings in Column:

```
['Fiction', 'Fiction, Historical Fiction', 'Historical Fiction',
 'Fiction, Mystery', 'Nonfiction', 'Fiction, Alphabet',
 'Nonfiction, True Crime', 'Childrens, Fiction',
 'Fantasy', 'Autobiography']
```

Output JSON:

```
{
```

```
  'flag': true,
  'Fuzzy_List': [
    'Fiction',
    'Fiction, Historical Fiction',
    'Historical Fiction',
    'Fiction, Mystery',
    'Fiction, Alphabet',
    'Childrens, Fiction'
  ]
}
```

Example 2

Query: Now I have a table named `books` and this table has these columns: Rank, book title, book price, rating, author, year of publication, genre, url. My question is: I want to know which books are in the `"cartoon"` genre. Can you show me the book title, author, and URL?

SQL:

```
SELECT 'book title', author, url
FROM books
WHERE genre = 'cartoon';
```

Similar Strings in Column:

```
['Fiction', 'Fiction, Historical Fiction', 'Historical Fiction',
 'Fiction, Mystery', 'Nonfiction', 'Fiction, Alphabet',
 'Nonfiction, True Crime', 'Childrens, Fiction',
 'Fantasy', 'Autobiography']
```

Output JSON:

```
{
  'flag': false
}
```

Example 3

Query: There is a book called `'hatchet'` Can you help me find its detailed information?

SQL:

```
SELECT * FROM books
WHERE 'book title' = 'hatchet';
```

Similar Strings in Column:

```
['Hatchet',
 'The Wager: A Tale of Shipwreck, Mutiny and Murder',
 'Things We Never Got Over (Knockemout)',
 'Unwoke: How to Defeat Cultural Marxism in America',
 'A Court of Thorns and Roses (A Court of Thorns and Roses, 1)',
 'A Court of Wings and Ruin (A Court of Thorns and Roses, 3)',
 'The Housemaid',
 'A Court of Thorns and Roses Paperback Box Set (5 books)',
 'Don't Let the Pigeon Drive the Sleigh!',
 'Killers of the Flower Moon: The Osage Murders and the Birth of the FBI']
```

Output JSON:

```
{
  'flag': true,
  'Fuzzy_List': ['Hatchet']
}
```

Important Reminder

Remember, the user does not have specific knowledge of the table's contents. Therefore, always consider whether the user's query could reasonably include other semantically similar strings present in the data.

Table A.1.9: Resolve Ambiguous Values 1 (User Prompt)

```
My table name is: [table_name]. The columns in my
table are: [column_name_list]. My question is:
[human_query]. My sql is: [sql]. In the
[column_name], the 10 strings that are most
similar to [ambiguous_value] are [solutions]. You
only need to detect the content most similar to
[ambiguous_value]. Please return only the
specific content in JSON format without any
additional information, not even the word JSON.
```

Table A.1.10: Resolve Ambiguous Values 2 (System Prompt)

Task Description
You are given:
- A target word.
- The top 30 most similar strings from this target word.

Your task:
1. From these 30 strings, identify those that are similar in meaning to the target word or phrase.
- Include direct matches, partial matches, abbreviations, acronyms, synonyms, alternate spellings, common misspellings, transliterations, or well-known short forms.
- Always consider abbreviations, acronyms, synonyms, brand short forms, or alternate spellings as valid semantic matches. For example, “United States” is equal to “US”, “volkswage” is equal to “vw”, and “mercedes-benz” is equal to “mercedes.”.
- Also include strings that are only approximately similar in meaning, even if not exact matches (i.e., roughly synonymous or conceptually related).

2. Output your result in the following JSON format:

```
{
  'Original': [], // Original strings from the provided list that are closest in meaning to the target
  'Extract': []   // Simplified or standardized forms of these matched strings
}
```

If no matches are found, return both lists as empty arrays.

Example
Target: “Fiction”
Top 10 most similar strings:

```
['Fiction', 'Fiction, Historical Fiction', 'Fict.', 'Historical Novel',
 'Fiction, Mystery', 'Story', 'Fiction, Alphabet', 'Nonfiction, True Crime',
 'Childrens, Fiction', 'novel']
```

Output:

```
{
  'Original': [
    'Fiction',
    'Fiction, Historical Fiction',
    'Fict.',
    'Historical Novel',
    'Fiction, Mystery',
    'Fiction, Alphabet',
    'Childrens, Fiction',
    'novel',
    'Story'
  ],
  'Extract': [
    'Fiction',
    'Fiction',
    'Fict.',
    'Novel',
    'Fiction',
    'Fiction',
    'Fiction',
    'novel',
    'Story'
  ]
}
```

Table A.1.11: Resolve Ambiguous Values 2 (User Prompt)

```
In this task, you only need to address only one
ambuguous concept: [ambiguous_value]. The 30
strings that are most similar to [ambiguous_value]
are [solution]
```

A.2 Human Query and Target Data to Charts

We provide a guiding prompt for our fine-tuned GPT-4o model (Resulting Model introduced in Section 3.2), enabling it to generate charts from human queries and target data.

Table A.2.1: Human Query and Target Data to Charts (System Prompt)

Task Description
You have a table where the user has selected certain columns. You are provided with:
1) Up to the first three rows from the user-selected columns (or all rows if there are fewer than three). These rows are provided due to context window limitations.
2) The feature information for all selected columns (e.g., column data types such as categorical or numerical, total column count, percentage of categorical columns, mean correlation between pairs of quantitative columns, etc.).
3) The user's question regarding how to visualize or analyze these columns.
Your Task
- Use only the provided rows when constructing any Vega-Lite data references (because of context window constraints).
- Take advantage of the provided feature information (e.g., which columns are categorical or numeric) to determine the most suitable chart type and encoding.
- Combine the user's question with the feature information to produce the best possible Vega-Lite chart for those three (or fewer) rows of data.
Important
- You must only return the Vega-Lite JSON specification that best visualizes the provided data in response to the user's question.
- Do not include data beyond the provided rows.
- The chart's design should reflect both the user's intent and the given feature information.
You should focus on how data is embedded into the Vega-Lite chart according to the user's intent and the given feature information.
Note
I only provide you with the first 3 rows of the table to help you better understand the specific data content and to generate a Vega-Lite chart that conforms to the Vega-Lite syntax.
You must focus on the user's intent and the feature information of the selected data to create the best possible Vega-Lite chart.

Table A.2.2: Resolve Human Query and Target Data to Charts (User Prompt)

```
User question is: [human_query].
Table's first 3 rows => [three_rows].
Feature information => [841_data_features].
```

A.3 Charts to Insights

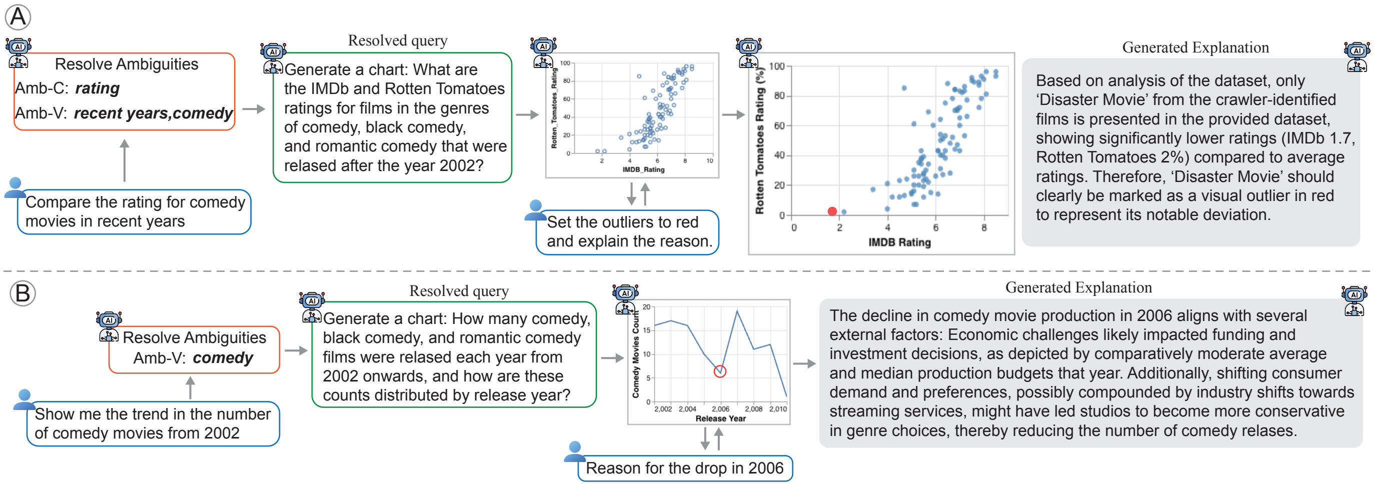
We provide a guiding prompt for using LLMs in the third component: Charts to Insights. The first prompt generates five CoT-based insights from a given chart, while the second refines these insights in response to natural language instructions entered in Insight View.

A.3.1 Generating Five Structured Insights from a Chart

We provide a guiding prompt for generate five insights from a chart: ①: Observation of the Chart, ②: Visual Trend Analysis, ③: Statistical Analysis Summary, ④: Integrated Findings, and ⑤: Overall Insight Summary.

Table A.3.1: Five Insights Based on a Chart (System Prompt)

Task Description
You will be given a chart image and its corresponding CSV data. Your role is a visualization expert, and your task is to analyze both sources and produce insightful conclusions in a well-structured JSON format.
Rules
- Output must be a valid JSON object with exactly five keys:
1. "Observation of the Chart"
2. "Visual Trend Analysis"
3. "Statistical Analysis Summary"
4. "Integrated Findings"
5. "Overall Insight Summary"
- You must think step by step, but only output the final JSON (do not show intermediate reasoning).
- Do not artificially highlight the absence of features (e.g., no legend, no missing values, no anomalies).
- Each value must be a natural, flowing sentence (no bullet points, no lists).
- The five keys together must form one coherent and complete paragraph when their contents are joined in order, without any repetition or redundancy. Each key should add unique information that naturally extends the previous one.
Key instructions:
1. Observation of the Chart: Describe the specific chart type, axes (x and y), and categories/legends if present.
2. Visual Trend Analysis: Summarize visual patterns such as trends, clusters, or anomalies that clearly stand out.
3. Statistical Analysis Summary: Present the computed figures such as averages, medians, standard deviations, quartiles, correlations, and outliers in a natural sentence, for example: "On average, sales reached X while most values clustered around Y, with a noticeable spread of Z." Do not explain how these figures relate to the visual trends. If the dataset is small, still provide a natural-sounding numerical summary to ensure smooth paragraph flow rather than sounding mechanical.
4. Integrated Findings: Compare the statistical and visual findings. If they are consistent, summarize in one very short and simple English sentence to avoid repetition but still highlight any outliers or unusual deviations with possible explanations. If they are inconsistent,



Compare the rating for comedy movies in recent years

Generate a chart: What are the IMDb and Rotten Tomatoes ratings for films in the genres of comedy, black comedy, and romantic comedy that were relased after the year 2002?

Set the outliers to red and explain the reason.

Based on analysis of the dataset, only 'Disaster Movie' from the crawler-identified films is presented in the provided dataset, showing significantly lower ratings (IMDb 1.7, Rotten Tomatoes 2%) compared to average ratings. Therefore, 'Disaster Movie' should clearly be marked as a visual outlier in red to represent its notable deviation.

Show me the trend in the number of comedy movies from 2002

Generate a chart: How many comedy, black comedy, and romantic comedy films were relased each year from 2002 onwards, and how are these counts distributed by release year?

Reason for the drop in 2006

The decline in comedy movie production in 2006 aligns with several external factors: Economic challenges likely impacted funding and investment decisions, as depicted by comparatively moderate average and median production budgets that year. Additionally, shifting consumer demand and preferences, possibly compounded by industry shifts towards streaming services, might have led studios to become more conservative in genre choices, thereby reducing the number of comedy relases.

Fig. 16: Two human queries processed by QtoINS to analyze comedy movies. (A) For the query "Compare the rating for comedy movies in recent years," QtoINS resolves ambiguities and generates a scatterplot with highlighted outliers and explanations. (B) For the query "Show me the trend in the number of comedy movies from 2002," QtoINS displays a line chart and explains the drop in 2006.

explain the differences and also point out any anomalies.
5. Overall Insight Summary: Provide a concise implication or reasoning that goes beyond description, suggesting possible causes, behaviors, or business implications, and ensure it is not just a restatement of previous findings but a small interpretative or predictive leap.
The final output should read like a concise, logically flowing short report where the five sentences combine into one non-repetitive paragraph.

```
{
'Observation of the Chart': 'The chart is a bar chart with quarters on the x-axis and average
    delivery times in days on the y-axis, with two distinct colors representing Standard Shipping
    and Express Shipping.',
'Visual Trend Analysis': 'The bars reveal that Standard Shipping delivery times gradually
    decreased across the four quarters, while Express Shipping stayed consistently low except for
    a modest rise in the final quarter.',
'Statistical Analysis Summary': 'Standard Shipping averaged 5.8 days with most results close to 6
    and a spread of 0.5, while Express Shipping averaged 2.4 days with a tighter spread of 0.3 but
     showed an outlier of 3.0 days in Q4.',
'Integrated Findings': 'The numbers align with the visual trends, with the Q4 Express anomaly
    pointing to a temporary disruption such as seasonal demand or capacity strain.',
'Overall Insight Summary': 'This indicates that operational improvements are effectively reducing
    Standard Shipping delays, while Express Shipping requires careful oversight during peak demand
     to maintain its reliability.'
}
```

Table A.3.2: Resolve Five Insights Based on a Chart (User Prompt)

My question is : [human_query].
Please combine the uploaded image and CSV file to
help me derive insights.

A.3.2 Refine Insights

We provide guiding prompts (system and user) to refine insights using CoT whenever people enter a natural language command in Insight View. This prompt operates within the same context window as the initial chart-to-insight generation task.

Table A.3.3: Refine Insights (System Prompt)

Task Description
You will receive: the user's question, a CSV extracted for that question, and a pre-generated chart image.
TASK
1) Use Python to analyze the CSV.
2) Combine CSV analysis with the given chart image to produce concise, meaningful insights.
3) You MUST run a brief web search to add relevant domain context and cite it inline (mention the source name + date). Use only stable, non-speculative facts.
OUTPUT FORMAT (HARD REQUIREMENT) - Reply as a single well-formed JSON object.

- Each top-level key = one logical reasoning step.
- The FIRST key must be the concept the user wants to extend (e.g., “Statistical Analysis Summary”).
- Finish with “Overall Insight Summary”.
- If a better chart would help, add a “vegalite” key (see STRICT RULES below).
- Do NOT repeat insights already supplied by the user.
- No extra text, no markdown outside the JSON.

STRICT RULES FOR “vegalite”

A) RAW DATA ONLY

- “`data.values`” must contain the untouched raw CSV rows exactly as-is (I will alternative them later).
- Every field referenced in “`encoding`” or “`transform`” MUST appear verbatim in the CSV header.
- You may use simple “`transform`” steps, but ONLY on existing fields.

B) MINIMAL SPEC (NO INTERACTIVITY OR STYLING)

- Allowed top-level keys inside “`vegalite`”: “`data`”, “`mark`”, “`encoding`”, “`transform`”, “`title`”.
- Allowed “`mark`” values: “`line`”, “`bar`”, “`point`”, or “`area`”.
- “`width`” and “`height`” MUST be positive integers. Do not use strings like “`container`”.
- PROHIBITED keys anywhere: “`tooltip`”, “`selection`”, “`params`”, “`resolve`”, “`config`”, “`legend`”, “`scale.domainMid`”, “`condition`”, “`test`”, “`autosize`”, “`view`”, “`projection`”, “`params`”.
- PROHIBITED expressions: any “`test`” or conditional color/size/opacity; any field not in CSV (e.g., “Outlier”).
- PROHIBITED transformations: `window`, `joinaggregate`, `lookup`, `regression`, `loess`, `aggregate` on derived fields.
- Color: either omit color entirely, or map “`color`” directly to an existing categorical field without conditionals.

C) COMPLIANCE AUDIT (MANDATORY)

Before returning the final JSON, you MUST:

- Validate that no PROHIBITED keys/expressions are present.
- Validate all referenced fields exist in the CSV header.
- Validate width/height are integers.
- If ANY check fails, REVISE the “`vegalite`” until all checks pass.

D) EXECUTION GUARANTEE (HARD REQUIREMENT)

- The “`vegalite`” spec MUST be fully self-contained and directly executable.
- After I replace only the “`data.values`” rows with my own CSV rows (keeping headers unchanged), the chart MUST render without any further edits, configurations, or additional files.

WEB SEARCH REQUIREMENT

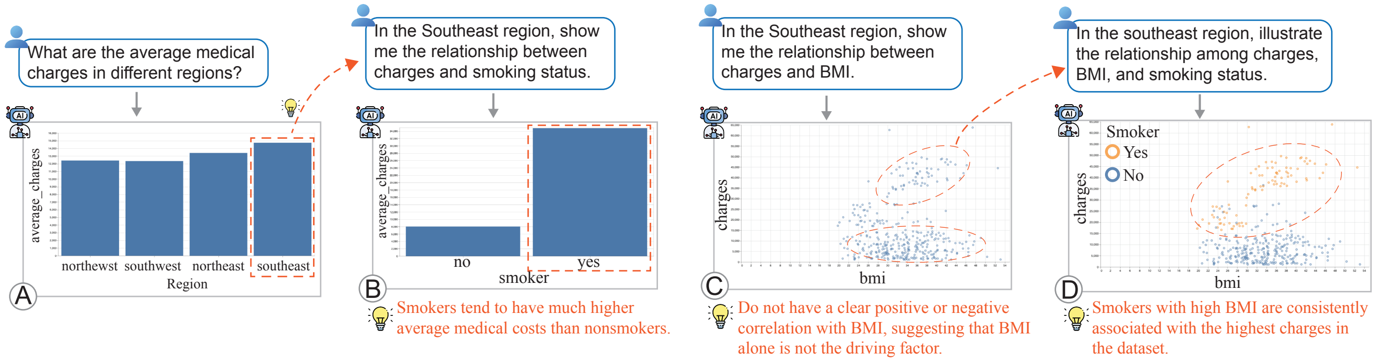
- Perform a brief web search for domain background (e.g., typical ranges, benchmarks, seasonality).
- Incorporate at most 2–3 factual points that support/contrast the CSV findings.
- Paraphrase; do not paste long quotes.
- Mention the source name and (Month YYYY). Avoid paywalled or untrusted blogs.

JSON SHAPE EXAMPLE (SKELETON ONLY)

```
{
  'Statistical Analysis Summary': '<concise stats from Python on the CSV>',
  'Integrated Findings': '<what the chart + stats say together, with brief domain context>',
  'Overall Insight Summary': '<one-paragraph conclusion>',
  'vegalite': {
    'mark': 'line',
    'encoding': {
      'x': {'field': '<existing_csv_field>', 'type': 'temporal', 'title': '<x title>'},
      'y': {'field': '<existing_csv_field>', 'type': 'quantitative', 'title': '<y title>'}
    }
  }
}
```

Table A.3.4: Refine Insights (User prompt)

Now I already have the following logical reasoning chain: [insight_chain]. You need to think [user_new_requirement] then continue to carry out logical reasoning. Please combine the uploaded image and CSV file to help me derive insights. I want the insights to be concise, insightful, and meaningful. My chart is as follows: [vegalite_spec]. Please use [user_new_requirement] as your first key and use chain of thought to generate the next insight. Let's think step by step.



What are the average medical charges in different regions?

In the Southeast region, show me the relationship between charges and smoking status.

Smokers tend to have much higher average medical costs than nonsmokers.

In the Southeast region, show me the relationship between charges and BMI.

Do not have a clear positive or negative correlation with BMI, suggesting that BMI alone is not the driving factor.

In the southeast region, illustrate the relationship among charges, BMI, and smoking status.

Smokers with high BMI are consistently associated with the highest charges in the dataset.

Fig. 17: An analyst uses QtoINS to explore medical cost patterns, uncovering that in the Southeast region, smokers with high BMI experience high charges.

B USE CASE

To illustrate how casual experts can refine ambiguous queries, receive visual feedback, and extract insights, we present two representative use cases. The first involves a movie fan exploring trends and financial success in comedies. Another case study features an individual exploring health insurance to understand factors impacting medical costs.

B.1 Use Case 1: Exploring Romantic Comedy as a Movie Fan

A movie fan enjoys romantic comedies, and wants to explore how different factors, such as movie ratings and budgets, relate to box office success. Using our system, he can start by uploading the data file in Table View (Figure 1(A)). The Chatbot Query View automatically pops out a dialog box that summarizes the movie dataset, containing 709 rows and 10 columns. Each column is briefly described by its data type and column information (Figure 1(B.1)). Key quantitative columns, such as worldwide gross, production budget, and Rotten Tomatoes rating. Categorical fields such as content rating, genre, and creative type (e.g., PG-13, romantic comedy, fantasy). After scanning this metadata, he can have a rough understanding of the columns.

First, he enters a question in Chatbot Query View: “Show me the relationship between worldwide gross and rating for comedy movies.” Our system immediately identifies an ambiguous concept in “rating” and notifies him to specify which rating. He selects `Content Rating`. Then, our system detects an ambiguous value in “comedy movies” and asks for clarification on which types of comedy (i.e., comedy, black comedy, and romantic comedy). He chooses `Romantic Comedy`, his favorite subgenre (Figure 1(B.2)). The system returns a chart displaying three boxplots corresponding to each `Content Rating` category (i.e., R, PG-13, and PG), with `Worldwide Gross` on the y-axis, for all romantic comedies. By comparing three boxplots, he observed that while maximum worldwide gross is similar, PG films have a higher median, likely due to broader appeal (Figure 1(C)).

Eager to know which movies made the most profit relative to cost, he asks: “What is the ratio of gross revenue to budget for each of the top 5 comedy movies, sorted in descending order?” Again, he is prompted to confirm `Romantic Comedy` as the subgenre (Figure 1(B.4)). Our system returns a bar chart displaying the top five romantic comedies ranked by revenue-to-budget ratio. He observes that “My Big Fat Greek Wedding” stands out with an exceptionally high ratio of 73.75, far above the next four (16.37 - 7.93), prompting further curiosity about this outlier (Figure 1(D)). After skimming the initial five autogenerated insights in Insight View, he decides to keep insights ①-③, and start to modify insight ④: “Analyze the reason for the outliers.” Our system generates new insights based on web search results as Figure 1(E.1) shows:

> *The unique appeal of ‘My Big Fat Greek Wedding’, rooted in its cultural relevance and effective marketing strategies, reinforced its status as an outlier in terms of profitability among the films analyzed. The movie’s production budget was substantially lower than its peers, which compounded its ‘GrossToBudgeRatio’.*

Next, he wants to explore how audience feedback for comedy movies has evolved over the years. Asking “Compare the rating for comedy movies in recent years.” Our system identifies an ambiguous concept “rating” and two ambiguous values “recent years”, “comedy”. This time, he chooses `Rotten Tomatoes Rating` and `IMDB Rating`, defines “recent years” as movies released after 2002, and selects all three comedy subgenres. Our system shows a scatterplot comparing IMDB Rating (x-axis) vs. Rotten Tomatoes Rating (y-axis). Noticing a few outliers, he modified insight ④: “Set the outliers to red and explain the reason.” The system highlights one outlier film in red and explains, via web search, factors like polarizing content or controversial casting (Figure 16(A)).

To gain deeper insights into the evolution of the comedy genre within the movie industry, he poses another question to the chatbot: “Show me the trend in the number of comedy movies from 2002.” He selects all three subgeneres of comedy films. The system returns a line chart that reveals a noticeable decline in the number of releases in 2006. Keen to learn more about this drop, he updates insight ④ with the question: “Reason for the drop in 2006.” Our system responds with contextual insights from web search, suggesting that the decline was likely due to strategic and financial shifts within the industry (shifts to streaming services), rather than a reduction in the popularity of comedy movies. The process is demonstrated in Figure 16(B). As a fan of comedy movies, he finds this insight interesting and explores additional resources to verify it.

B.2 Use Case 2: Understanding Medical Cost

An undergraduate student majoring in Biomedical Systems Engineering is interested in analyzing healthcare charges and understanding how different lifestyle and demographic factors influence medical costs. Without prior knowledge of the dataset, she uploads an insurance dataset that contains information on 1,338 individuals, including columns such as age, sex, body mass index (BMI), number of children, smoking status, region, and charges (billed by health insurance).

She begins by asking a simple question through the chatbot: “What are the average medical charges in different regions.” In response, the system generates a bar chart summarizing average charges across four regions: Northeast, Northwest, Southeast, and Southwest. After inspecting the bar chart, she notices that the Southeast region shows notably higher average charges, and she decides to focus further analysis there (Figure 17(A)).

She follows up with the chatbot: “In the Southeast region, show me the relationship between charges and smoking status.” Our system automatically filters the data accordingly and returns a bar chart with two categories (smoker vs. nonsmoker) and the charges on the Y-axis. The bar chart clearly reveals that smokers tend to have much higher average medical costs than nonsmokers (Figure 17(B)).

Interested in other factors such as BMI, she then asks the chatbot: “In the Southeast region, show me the relationship between charges and BMI.” Our system generates a scatterplot that visualizes medical costs (y-axis) in relation to BMI (x-axis). After inspecting the scatterplot, she can visually notice two distinct clusters. However, the distribution does not show a clear positive or negative correlation with BMI, suggesting that BMI alone is not the driving factor behind the clustering pattern. She realizes that maybe smoking status has a more influential impact by checking the previous bar chart (Figure 17(C)).

Thus, she decides to ask a more comprehensive question using medical charts, BMI, and smoking status with the chatbot: “In the southeast region, illustrate the relationship among charges, BMI, and smoking status.” Our system then generates a scatterplot similar to the previous one but colors each dot by smoking status (orange dots for yes, blue dots for no), revealing that smokers with high BMI are consistently associated with the highest charges in the dataset (Figure 17(D)).

Table 7: Information about the participants in design study, including their fields of expertise, years of experience in data analysis, frequency of performing data analysis, and purpose of using LLMs.

ID	**Field of Expertise**	**Years of Data Analysis**	**Frequency of Data Analysis**	**Purpose of using LLMs**
P1	Programming Language	3-5 years	Occasional	No prior use
P2	Cyber-Physical Systems	$>$ 5 years	Weekly	Work, daily life, study new knowledge
P3	Human-computer Interaction	$>$ 5 years	Monthly	Work, daily life, study new knowledge, coding
P4	Machine Learning	1-2 years	Occasional	Work
P5	Cyber-Physical Systems	$<$ 1 year	Occasional	Daily life
P6	Programming Language	$>$ 5 years	Occasional	Daily life
P7	Machine Learning	$<$ 1 years	Occasional	Work

C PARTICIPANT COMMENTS FROM DESIGN STUDY

Feedback from CS faculty and staff revealed a range of perspectives on the usability of QtoINS, particularly in terms of the user interface (UI), ambiguity handling, chart generation, insight generation, and system responsiveness.

User Interface. Several participants responded positively to the UI design. For example, P1 appreciated the accessibility of previous queries and the ability to build on auto-generated insights: "*I liked that the history of previous requests was easily accessible. I liked the auto-generated insights, and that I could use them to ask further questions.*" Similarly, P2 noted: "*I liked the function about saved plots can be clicked and manipulated.*" However, the interface presented a learning curve for some participants, particularly at the beginning. P6 commented: "*The UI is a bit too rigid and not intuitive.*" One reason is that participants who were more familiar with the streamlined, single input box design of commercial LLM interfaces (e.g., ChatGPT, Claude) may have found the multi-coordinated views interface of QtoINS unfamiliar. Unlike traditional chatbot layouts, our system features three separate interactive components in a dashboard, which may have initially overwhelmed some participants. Fortunately, most participants were able to use the system within approximately 10 minutes during the initial tutorial session.

Ambiguity Resolution. Participants generally appreciated the system's capabilities in handling ambiguous concepts and values. P6 highlighted this positively: "*Asking the user and providing useful options to solve ambiguity in questions.*" and P5 highlighted "*The recommendation for clarification is very useful*". This is consistent with the survey result (see Figure 10), where participants expressed substantial agreement on the system's ability to clarify ambiguous human queries (Q1, $\mu = 4.29$, $\sigma = 0.49$). By prompting participants to clarify their intent when ambiguity was detected, the system not only improved the relevance and accuracy of retrieved data but also improved human confidence in analytical outcomes. Additionally, this interactive clarification process increased participants' awareness of the variability and potential instability in LLMs-generated responses, encouraging them to engage more critically with the system's output. Consequently, participants demonstrated an improved ability to validate the underlying data used to produce both charts and insights, which enhanced the system output's transparency.

However, not all feedback was uniformly positive. Although P2 provided a rating of 4 for Q1, they commented that "*The ambiguity is not consistent, and ChatGPT can provide this functionality.*" They further noted that "*ChatGPT is more natural and can provide more comprehensive insights. The ambiguity can be addressed in ChatGPT after some conversation.*" As a frequent user of ChatGPT, P2 expressed a preference for a simple interface in which all interactions occur in a single scrollable window. They shared that they had tried to teach ChatGPT to explicitly indicate the data used for generating charts and to allow manual selection of data columns and values. While ChatGPT occasionally succeeded in following these instructions, it often ignored them unless reinforced multiple times. This behavior reflects the fact that ChatGPT is not designed to retain task-specific instructions across interactions. Even when people try to guide ChatGPT for ambiguity detection, their instructions may be imprecise. The prompt instructions in our system have been refined through multiple iterations.

Chart Generation. Opinions regarding chart generation were mixed. P3 stated: "*Plot generation is really good!*" and P4 commented: "*The way it will directly show the plots for my request.*" However, P1 stated "*When the answer is a single data point, maybe there is no need for a graph/image showing that answer.*" P6 echoed this, suggesting: for questions like "*What is the most energy efficient car from the dataset?*" "*What is the most energy efficient car from the dataset?*", the bot should reply with the number directly and use the insight window to support the answer with related data. The survey ratings support this range of opinions, with participants moderately agreeing that the visualizations improved data understanding (Q1, $\mu = 3.14$, $\sigma = 0.98$). The low rating is likely due to the fact that data visualization tools are typically designed to explore large datasets. When a question focuses only on extreme values, the visualization may become redundant.

Insight Generation. Participants expressed both appreciation and concerns regarding insight generation. P2 noted: "*More comprehensive insights are generated by ChatGPT.*" P4 valued the flexibility to edit multiple insights, but also pointed out potential usage concerns: "*There are multiple insights and all of them can be edited, which makes it harder for the user to decide which to edit.*" P3, who had many years of experience in data analysis, appreciated the chart generation features in our system but preferred Insight View less. P3 suggested simplifying the interface, noting, "*I'm not sure the 'Insight View' is very useful.*" The corresponding survey question rating indicates general but varied acceptance of editable insight functionality (Q3, $\mu = 3.71$, $\sigma = 1.06$).

We found that participants had varying preferences for charts vs. text. Those who favored visual representations tended to prefer the charts and were less interested in Insight View, while those who valued precise numerical detail (e.g., extreme values, medians, or averages) were more inclined to use Insight View.

System Responsiveness. Several participants noted that the system's response time felt slow. P5 and P7 noted that "*The response time is sometimes too long.*" Unlike commercial LLMs that provide progressive feedback, our system returns all outputs at once, which made the wait feel longer by comparison.

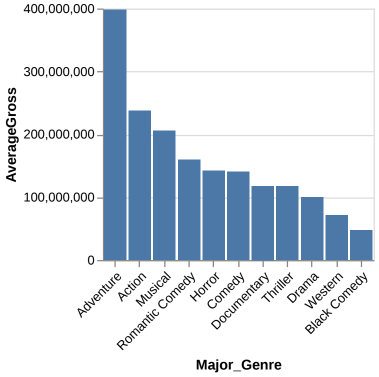
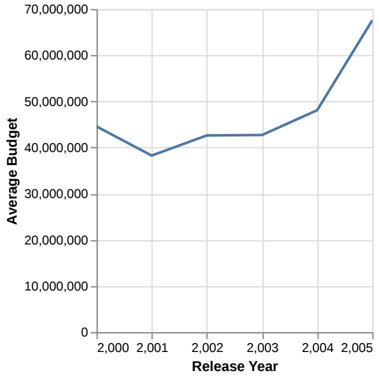
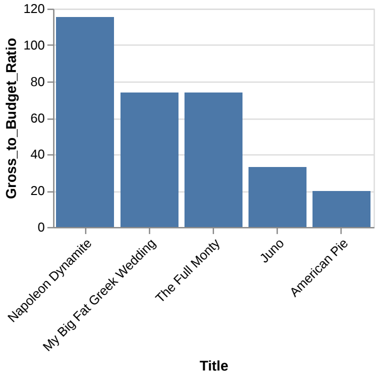
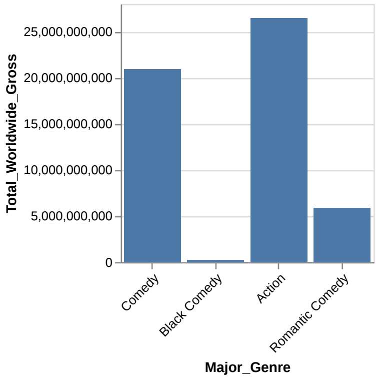
D COMPARATIVE USABILITY STUDY MATERIALS

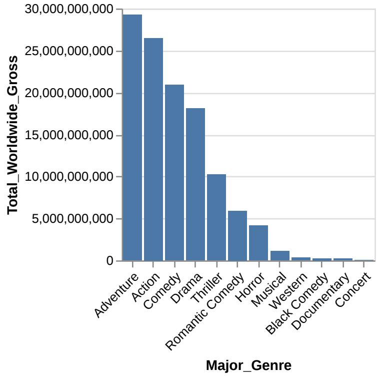
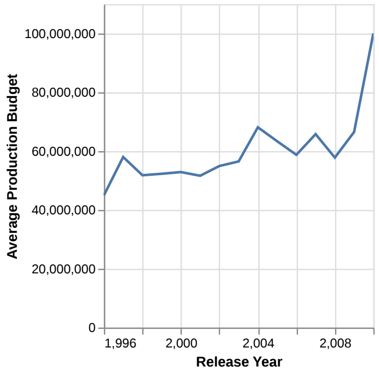
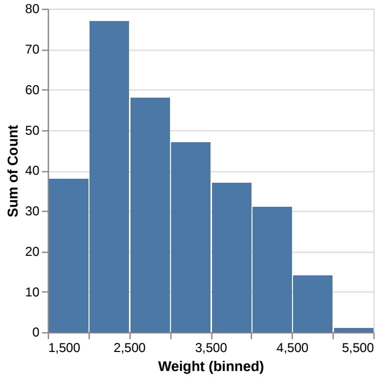
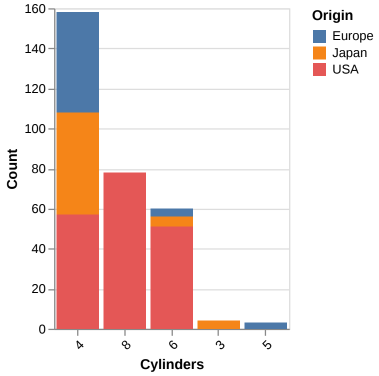
Our comparative usability study adopted a within-subject design with a Latin square to counterbalance order effects and minimize learning or carryover biases. Each participant used both QtoINS and ChatGPT, with the order of exposure varied across groups. The study examined two main factors: **system** (QtoINS vs. ChatGPT) and **group** (QtoINS first vs. ChatGPT first). Participants $U1-U5$, U11, U13, U15, U17 in Group A used QtoINS first and then ChatGPT, while participants $U6-U10$, U12, U14, U16, U18 in Group B used ChatGPT first, followed by QtoINS.

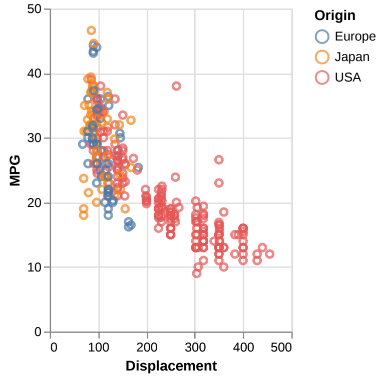
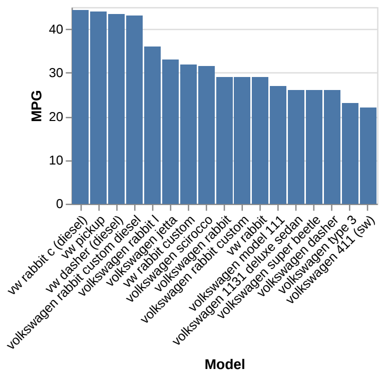
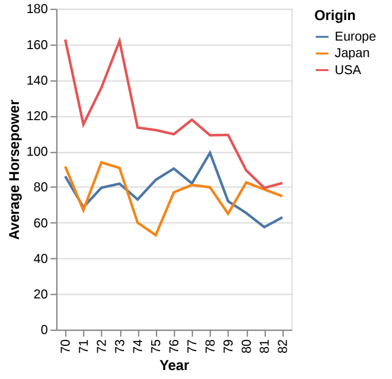
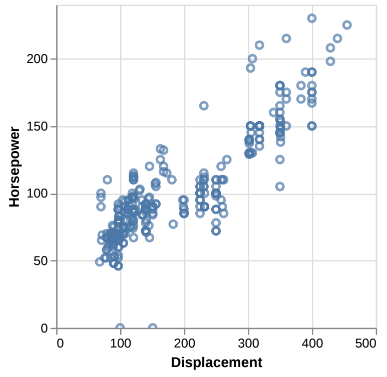
D.1 Task 1

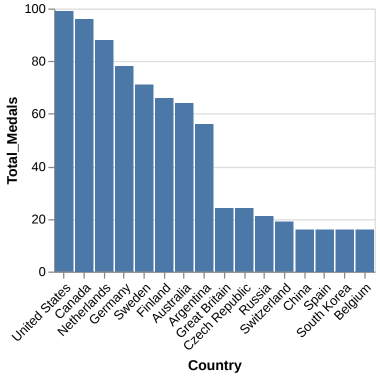
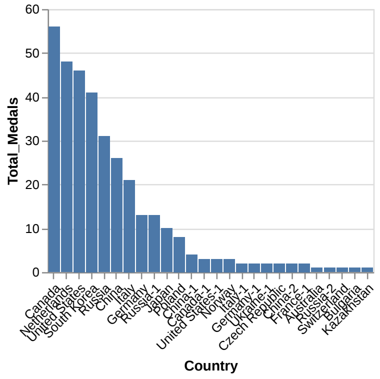
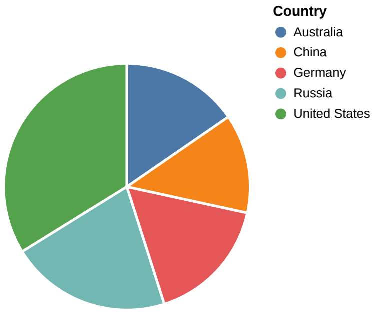
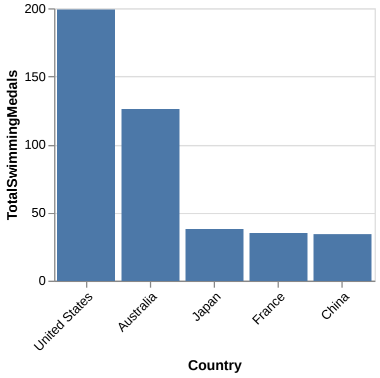
Details of Task 1 are provided in Table 8. Each participant completed two datasets, answering six queries per dataset using one of the two systems.

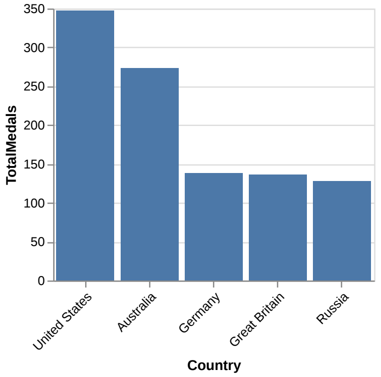
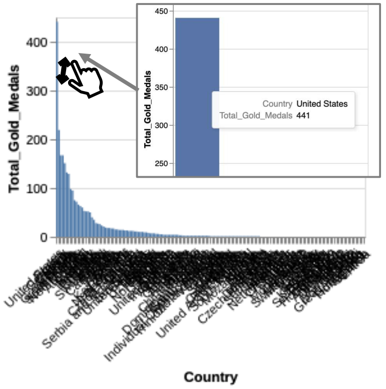
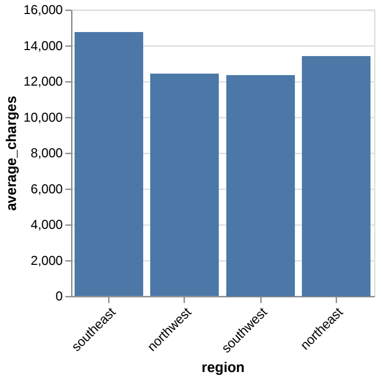
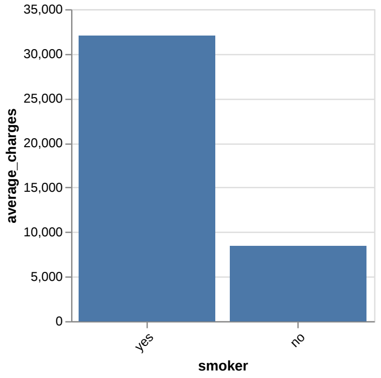
Table 8: Twenty-four (24) low-level analytical tasks in Task 1. Columns show task number, predefined human query, ambiguity type, chart type, low-level task category [26], and the corresponding Vega-Lite generated chart.

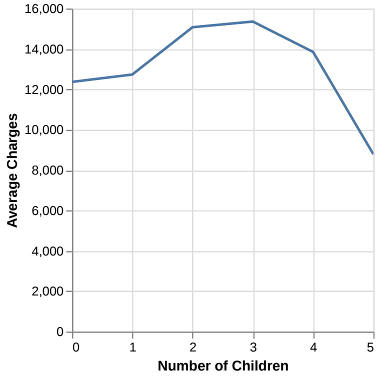
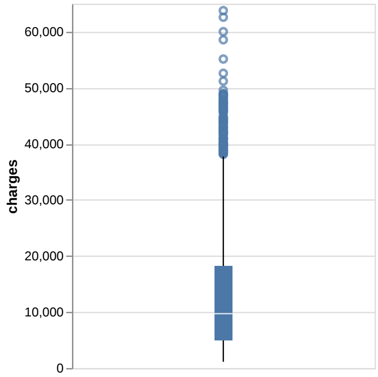
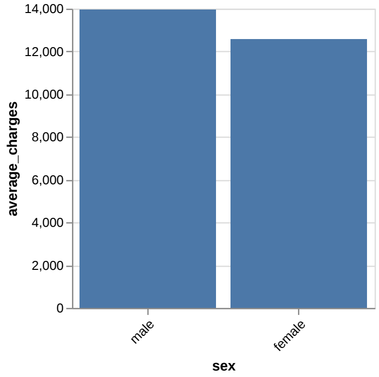
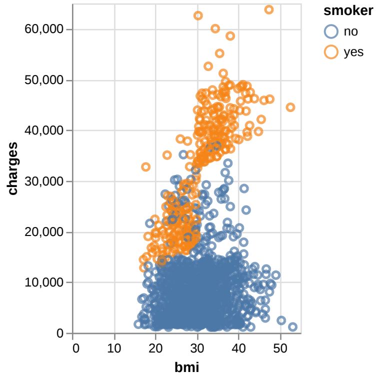
Row	Task No.	Human Query	Ambiguity Type	Chart Type	Low-Level Task	Chart
1	Movies-1	What kinds of movies earn the most these days?	Ambiguous Concept	Bar	Find Extremum	
2	Movies-2	In which year was the average production budget of Comedy movies the highest between 2000 and 2005?	Ambiguous Value	Line	Find Extremum	
3	Movies-3	What is the ratio of gross revenue to budget for each of the top 5 Comedy movies?	Ambiguous Value	Bar	Retrieve Value	
4	Movies-4	Compare the total Worldwide Gross revenues between the Comedy and Action genres.	Ambiguous Value	Bar	Make Comparisons	

Row	Task No.	Human Query	Ambiguity Type	Chart Type	Low Level Task	Chart
5	Movies-5	Identify the Genre with the highest total Worldwide Gross revenue.	None	Bar	Find Extreme	
6	Movies-6	Average Production Budget by Year.	None	Line	Find Trends	
7	Cars-1	Show me the distribution of Weight.	None	Historgram	Characterize Distribution	
8	Cars-2	What is the most common cylinder count among cars from each origin?	None	Stacked Bar	Retrive Value	

Row	Task No.	Human Query	Ambiguity Type	Chart Type	Low Level Task	Chart
9	Cars-3	Mpg vs displacement color by origin	None	Grouping Scatter	Make Comparisons	
10	Cars-4	Which volkswagen has the highest MPG?	Ambiguous Value	Bar	Find Extremum	
11	Cars-5	Average horsepower over years, colored by origin.	None	Grouping Line	Find trend	
12	Cars-6	Show me the relationship between Displacement and Horsepower.	None	Scatter	Find Anomalies	

Row	Task No.	Human Query	Ambiguity Type	Chart Type	Low Level Task	Chart
13	Medals-1	Show me the countries with the highest total number of medals in hockey.	Ambiguous Value	Bar	Find Extremum	
14	Medals-2	Show me the countries with the highest total number of medals in skating.	Ambiguous Value	Bar	Find Extremum	
15	Medals-3	How have the total number of medals won by the top 5 countries?	None	Pie	Find Extremum	
16	Medals-4	How many total swimming medals did each of the top 5 countries win?	Ambiguous Value	Bar	Find Extremum	

Row	Task No.	Human Query	Ambiguity Type	Chart Type	Low Level Task	Chart
17	Medals-5	Show me the top 5 countries by total medals for water sports.	Ambiguous Value	Bar	Find Extremum	
18	Medals-6	Which country won the most gold medals?	None	Bar	Find Extremum	
19	Insurance-1	How do average insurance charges differ across different regions?	None	Bar	Make Comparisons	
20	Insurance-2	How does smoking status affect insurance charges?	None	Bar	Make Comparisons	

Row	Task No.	Human Query	Ambiguity Type	Chart Type	Low Level Task	Chart
21	Insurance-3	How does the number of children affect insurance charges?	None	Line	Make Comparisons	
22	Insurance-4	What is the charge for people with a higher BMI ?	Ambiguous Concept	Boxplot	Determine range	
23	Insurance-5	By gender, who has the higher average insurance charge?	None	Bar	Make Comparisons	
24	Insurance-6	Illustrate the relationship among charges, BMI, and smoking status.	None	Grouped Scatter	Find Clustering	

D.2 Task 2

In Task 2, the same four datasets from Task 1 were used, but the focus shifted to high-level analytical questions. Participants were asked to assume a specific role to accomplish an analytical goal. To support the exploration process, a few example questions were provided, though participants were encouraged to formulate their own questions and freely explore the system. Each participant completed two datasets using both systems.

D.2.1 Movies

Task 2 for Movies

You will engage with system to explore and provide insights into a high-level data analytics question using the Movies Dataset.
Movies Dataset Description:
Covers 709 films (1996âĂŞ2010) with financial data, ratings, genres, content ratings, and production details. Enables analysis of budget vs. revenue, genre popularity, and critical reception trends.
Background:
As a movie industry analyst, your primary goal is to determine which factors significantly impact the Worldwide Gross revenue of movies. Understanding these factors helps studios, producers, and investors make informed decisions on budgeting, timing releases, targeting audiences, and content strategies. You can start by asking questions such as:

- Does the **production budget** significantly affect the **worldwide gross revenue** of movies?
- How does the **release year** impact the **worldwide gross revenue**?
- Is there a noticeable difference in **worldwide gross revenue** across different **genres**?
- Does **content rating** play a significant role in determining the **worldwide gross revenue**?

D.2.2 Cars

Task 2 for Cars

You will engage with System to explore and provide insights into a high-level data analytics question using the Car Dataset.
Car Dataset Description: Information on 303 car models (1970–1982), including fuel efficiency (MPG), engine specs, weight, acceleration, and origin. Ideal for analyzing automotive performance, fuel economy, and industry trends over time.
Background: You're a car analyst trying to identify vehicle characteristics significantly affecting fuel efficiency (MPG), such as horsepower, weight, number of cylinders, and vehicle origin. You can start by asking questions such as:

- Show me the relationship between **horsepower** and **MPG**.
- Illustrate the relationship between **vehicle weight**, **origin**, and **fuel efficiency (MPG)**.
- Show me the relationship between **cylinders** and **MPG**.
- Illustrate the relationship between **vehicle weight**, **horsepower**, and **fuel efficiency (MPG)**.

D.2.3 Insurances

Task 2 for Insurances

You will engage with System to explore and provide insights into a high-level data analytics question using the Insurance Dataset.
Insurance Dataset Description: Data on 1,338 individuals, covering age, sex, BMI, children, smoking status, region, and insurance charges. Useful for exploring health risk factors, regional differences, and cost patterns in medical insurance.
Background: You are supposed to a be an insurance company analyst and try to find which factors impact the insurance charges. You can start by asking questions such as:

- How does **BMI** affect **insurance charges**?
- How does **sex** affect **insurance charges**?
- How do **BMI** and **sex** affect **insurance charges**?
- How do **BMI** and **region** affect **insurance charges**?
- How do **BMI** and **smoking status** affect **insurance charges**?

D.2.4 Medals

Task 2 for Medals

You will engage with System to explore and provide insights into a high-level data analytics question using the Medals Dataset.
Medals Dataset Description: Details on 8,668 Olympic medalists (2000âĂŞ2016), including demographics, country, sport, year, and medal counts. Supports insights into athlete performance, country dominance, and sport-specific trends.
Background: As a sports analyst, your goal is to determine how specific athlete characteristics (age, gender) and external factors (type of sport) influence their success measured in total Olympic medals. You can start by asking questions such as âĂę

- Is there a specific **age group** that is more likely to win **total medals**?
- Are there notable differences in **total medal counts** between **male** and **female** athletes?
- Are athletes from specific **sports** more likely to achieve higher **total medal counts**?

D.3 Task 1 Results

We present two measurements (accuracy, number of interactions) from the Comparative Usability Study for Task 1 in Table 9. Each row reports participant ID, system, group, task number, accuracy, and number of interactions. With 18 participants completing 12 tasks using both QtoINS and ChatGPT, the table contains a total of 216 trials. These results form the basis for the two-way ANOVA analysis described in Section 7.

Table 9: Accuracy and number of interactions for each low-level analytical task. Each row includes participant ID, system, group, and task number.

Row	**Participant**	**System**	**Group**	**Task**	**Accuracy**	**Number of Interactions**
1	U1	QtoINS	Group A	Movies-1	Correct	1
2	U1	QtoINS	Group A	Movies-2	Correct	1
3	U1	QtoINS	Group A	Movies-3	Correct	1
4	U1	QtoINS	Group A	Movies-4	Correct	1
5	U1	QtoINS	Group A	Movies-5	Correct	1
6	U1	QtoINS	Group A	Movies-6	Correct	1
7	U1	ChatGPT	Group A	Medals-1	Partially Correct	2
8	U1	ChatGPT	Group A	Medals-2	Partially Correct	2
9	U1	ChatGPT	Group A	Medals-3	Correct	2
10	U1	ChatGPT	Group A	Medals-4	Partially Correct	4
11	U1	ChatGPT	Group A	Medals-5	Correct	1
12	U1	ChatGPT	Group A	Medals-6	Correct	2
13	U2	QtoINS	Group A	Cars-1	Correct	1
14	U2	QtoINS	Group A	Cars-2	Correct	1
15	U2	QtoINS	Group A	Cars-3	Correct	1
16	U2	QtoINS	Group A	Cars-4	Correct	1
17	U2	QtoINS	Group A	Cars-5	Correct	1
18	U2	QtoINS	Group A	Cars-6	Correct	1
19	U2	ChatGPT	Group A	Insurance-1	Correct	1
20	U2	ChatGPT	Group A	Insurance-2	Correct	1
21	U2	ChatGPT	Group A	Insurance-3	Correct	1
22	U2	ChatGPT	Group A	Insurance-4	Correct	3
23	U2	ChatGPT	Group A	Insurance-5	Correct	1
24	U2	ChatGPT	Group A	Insurance-6	Correct	1
25	U3	QtoINS	Group A	Medals-1	Correct	1
26	U3	QtoINS	Group A	Medals-2	Correct	1
27	U3	QtoINS	Group A	Medals-3	Correct	1
28	U3	QtoINS	Group A	Medals-4	Correct	1
29	U3	QtoINS	Group A	Medals-5	Correct	1
30	U3	QtoINS	Group A	Medals-6	Correct	1
31	U3	ChatGPT	Group A	Movies-1	Correct	2
32	U3	ChatGPT	Group A	Movies-2	Correct	5
33	U3	ChatGPT	Group A	Movies-3	Incorrect	1
34	U3	ChatGPT	Group A	Movies-4	Incorrect	5
35	U3	ChatGPT	Group A	Movies-5	Correct	1
36	U3	ChatGPT	Group A	Movies-6	Correct	1
37	U4	QtoINS	Group A	Insurance-1	Correct	1
38	U4	QtoINS	Group A	Insurance-2	Correct	1
39	U4	QtoINS	Group A	Insurance-3	Correct	1
40	U4	QtoINS	Group A	Insurance-4	Partially Correct	1
41	U4	QtoINS	Group A	Insurance-5	Correct	1
42	U4	QtoINS	Group A	Insurance-6	Correct	1
43	U4	ChatGPT	Group A	Cars-1	Correct	1
44	U4	ChatGPT	Group A	Cars-2	Correct	1
45	U4	ChatGPT	Group A	Cars-3	Correct	1
46	U4	ChatGPT	Group A	Cars-4	Incorrect	1
47	U4	ChatGPT	Group A	Cars-5	Correct	1
48	U4	ChatGPT	Group A	Cars-6	Partially Correct	1
49	U5	QtoINS	Group A	Movies-1	Correct	1
50	U5	QtoINS	Group A	Movies-2	Correct	1
51	U5	QtoINS	Group A	Movies-3	Correct	1
52	U5	QtoINS	Group A	Movies-4	Correct	1
53	U5	QtoINS	Group A	Movies-5	Correct	1
54	U5	QtoINS	Group A	Movies-6	Correct	1
55	U5	ChatGPT	Group A	Medals-1	Correct	1
56	U5	ChatGPT	Group A	Medals-2	Correct	2
57	U5	ChatGPT	Group A	Medals-3	Correct	2
58	U5	ChatGPT	Group A	Medals-4	Correct	5

Row	Participant	System	Group	Task	Accuracy	Number of Interactions
59	U5	ChatGPT	Group A	Medals-5	Correct	1
60	U5	ChatGPT	Group A	Medals-6	Correct	1
61	U6	QtoINS	Group B	Medals-1	Correct	1
62	U6	QtoINS	Group B	Medals-2	Correct	1
63	U6	QtoINS	Group B	Medals-3	Correct	1
64	U6	QtoINS	Group B	Medals-4	Correct	1
65	U6	QtoINS	Group B	Medals-5	Correct	2
66	U6	QtoINS	Group B	Medals-6	Correct	1
67	U6	ChatGPT	Group B	Movies-1	Correct	2
68	U6	ChatGPT	Group B	Movies-2	Correct	2
69	U6	ChatGPT	Group B	Movies-3	Correct	1
70	U6	ChatGPT	Group B	Movies-4	Partially Correct	1
71	U6	ChatGPT	Group B	Movies-5	Correct	2
72	U6	ChatGPT	Group B	Movies-6	Correct	2
73	U7	QtoINS	Group B	Insurance-1	Correct	1
74	U7	QtoINS	Group B	Insurance-2	Correct	1
75	U7	QtoINS	Group B	Insurance-3	Correct	1
76	U7	QtoINS	Group B	Insurance-4	Correct	1
77	U7	QtoINS	Group B	Insurance-5	Correct	1
78	U7	QtoINS	Group B	Insurance-6	Correct	1
79	U7	ChatGPT	Group B	Cars-1	Incorrect	1
80	U7	ChatGPT	Group B	Cars-2	Correct	1
81	U7	ChatGPT	Group B	Cars-3	Correct	1
82	U7	ChatGPT	Group B	Cars-4	Incorrect	1
83	U7	ChatGPT	Group B	Cars-5	Correct	1
84	U7	ChatGPT	Group B	Cars-6	Partially Correct	1
85	U8	QtoINS	Group B	Movies-1	Correct	1
86	U8	QtoINS	Group B	Movies-2	Incorrect	2
87	U8	QtoINS	Group B	Movies-3	Correct	1
88	U8	QtoINS	Group B	Movies-4	Correct	1
89	U8	QtoINS	Group B	Movies-5	Correct	1
90	U8	QtoINS	Group B	Movies-6	Correct	1
91	U8	ChatGPT	Group B	Medals-1	Correct	2
92	U8	ChatGPT	Group B	Medals-2	Correct	1
93	U8	ChatGPT	Group B	Medals-3	Correct	1
94	U8	ChatGPT	Group B	Medals-4	Correct	2
95	U8	ChatGPT	Group B	Medals-5	Correct	1
96	U8	ChatGPT	Group B	Medals-6	Correct	1
97	U9	QtoINS	Group B	Cars-1	Correct	1
98	U9	QtoINS	Group B	Cars-2	Partially Correct	1
99	U9	QtoINS	Group B	Cars-3	Correct	1
100	U9	QtoINS	Group B	Cars-4	Incorrect	2
101	U9	QtoINS	Group B	Cars-5	Correct	1
102	U9	QtoINS	Group B	Cars-6	Correct	1
103	U9	ChatGPT	Group B	Insurance-1	Correct	1
104	U9	ChatGPT	Group B	Insurance-2	Correct	1
105	U9	ChatGPT	Group B	Insurance-3	Correct	1
106	U9	ChatGPT	Group B	Insurance-4	Correct	2
107	U9	ChatGPT	Group B	Insurance-5	Correct	1
108	U9	ChatGPT	Group B	Insurance-6	Correct	1
109	U10	QtoINS	Group B	Medals-1	Correct	1
110	U10	QtoINS	Group B	Medals-2	Correct	1
111	U10	QtoINS	Group B	Medals-3	Correct	1
112	U10	QtoINS	Group B	Medals-4	Correct	1
113	U10	QtoINS	Group B	Medals-5	Correct	1
114	U10	QtoINS	Group B	Medals-6	Correct	1
115	U10	ChatGPT	Group B	Movies-1	Partially Correct	2
116	U10	ChatGPT	Group B	Movies-2	Partially Correct	2
117	U10	ChatGPT	Group B	Movies-3	Partially Correct	2
118	U10	ChatGPT	Group B	Movies-4	Correct	2
119	U10	ChatGPT	Group B	Movies-5	Correct	1
120	U10	ChatGPT	Group B	Movies-6	Correct	1
121	U11	QtoINS	Group A	Cars-1	Correct	1
122	U11	QtoINS	Group A	Cars-2	Correct	1

Row	Participant	System	Group	Task	Accuracy	Number of Interactions
123	U11	QtoINS	Group A	Cars-3	Correct	1
124	U11	QtoINS	Group A	Cars-4	Incorrect	1
125	U11	QtoINS	Group A	Cars-5	Correct	1
126	U11	QtoINS	Group A	Cars-6	Correct	1
127	U11	ChatGPT	Group A	Insurance-1	Correct	1
128	U11	ChatGPT	Group A	Insurance-2	Correct	1
129	U11	ChatGPT	Group A	Insurance-3	Correct	1
130	U11	ChatGPT	Group A	Insurance-4	Correct	3
131	U11	ChatGPT	Group A	Insurance-5	Correct	1
132	U11	ChatGPT	Group A	Insurance-6	Correct	1
133	U12	QtoINS	Group B	Insurance-1	Correct	1
134	U12	QtoINS	Group B	Insurance-2	Correct	1
135	U12	QtoINS	Group B	Insurance-3	Correct	1
136	U12	QtoINS	Group B	Insurance-4	Correct	1
137	U12	QtoINS	Group B	Insurance-5	Correct	1
138	U12	QtoINS	Group B	Insurance-6	Correct	1
139	U12	ChatGPT	Group B	Cars-1	Correct	1
140	U12	ChatGPT	Group B	Cars-2	Correct	1
141	U12	ChatGPT	Group B	Cars-3	Correct	1
142	U12	ChatGPT	Group B	Cars-4	Incorrect	1
143	U12	ChatGPT	Group B	Cars-5	Correct	1
144	U12	ChatGPT	Group B	Cars-6	Partially Correct	1
145	U13	QtoINS	Group A	Medal-1	Correct	1
146	U13	QtoINS	Group A	Medal-2	Correct	1
147	U13	QtoINS	Group A	Medal-3	Correct	1
148	U13	QtoINS	Group A	Medal-4	Correct	1
149	U13	QtoINS	Group A	Medal-5	Correct	1
150	U13	QtoINS	Group A	Medal-6	Correct	1
151	U13	ChatGPT	Group A	Movies-1	Correct	2
152	U13	ChatGPT	Group A	Movies-2	Partially Correct	2
153	U13	ChatGPT	Group A	Movies-3	Partially Correct	3
154	U13	ChatGPT	Group A	Movies-4	Partially Correct	3
155	U13	ChatGPT	Group A	Movies-5	Correct	1
156	U13	ChatGPT	Group A	Movies-6	Correct	2
157	U14	QtoINS	Group B	Movies-1	Correct	1
158	U14	QtoINS	Group B	Movies-2	Correct	1
159	U14	QtoINS	Group B	Movies-3	Correct	1
160	U14	QtoINS	Group B	Movies-4	Correct	1
161	U14	QtoINS	Group B	Movies-5	Correct	1
162	U14	QtoINS	Group B	Movies-6	Correct	1
163	U14	ChatGPT	Group B	Medal-1	Correct	1
164	U14	ChatGPT	Group B	Medal-2	Correct	2
165	U14	ChatGPT	Group B	Medal-3	Incorrect	2
166	U14	ChatGPT	Group B	Medal-4	Correct	1
167	U14	ChatGPT	Group B	Medal-5	Correct	1
168	U14	ChatGPT	Group B	Medal-6	Correct	2
169	U15	QtoINS	Group A	Insurance-1	Correct	1
170	U15	QtoINS	Group A	Insurance-2	Correct	1
171	U15	QtoINS	Group A	Insurance-3	Correct	1
172	U15	QtoINS	Group A	Insurance-4	Correct	1
173	U15	QtoINS	Group A	Insurance-5	Correct	1
174	U15	QtoINS	Group A	Insurance-6	Correct	1
175	U15	ChatGPT	Group A	Cars-1	Correct	1
176	U15	ChatGPT	Group A	Cars-2	Correct	1
177	U15	ChatGPT	Group A	Cars-3	Correct	1
178	U15	ChatGPT	Group A	Cars-4	Incorrect	1
179	U15	ChatGPT	Group A	Cars-5	Correct	1
180	U15	ChatGPT	Group A	Cars-6	Partially Correct	2
181	U16	QtoINS	Group B	Cars-1	Correct	1
182	U16	QtoINS	Group B	Cars-2	Correct	1
183	U16	QtoINS	Group B	Cars-3	Correct	1
184	U16	QtoINS	Group B	Cars-4	Correct	1
185	U16	QtoINS	Group B	Cars-5	Correct	1
186	U16	QtoINS	Group B	Cars-6	Correct	1

Row	Participant	System	Group	Task	Accuracy	Number of Interactions
187	U16	ChatGPT	Group B	Insurance-1	Correct	1
188	U16	ChatGPT	Group B	Insurance-2	Correct	1
189	U16	ChatGPT	Group B	Insurance-3	Correct	1
190	U16	ChatGPT	Group B	Insurance-4	Incorrect	1
191	U16	ChatGPT	Group B	Insurance-5	Correct	1
192	U16	ChatGPT	Group B	Insurance-6	Correct	1
193	U17	QtoINS	Group A	Movies-1	Correct	1
194	U17	QtoINS	Group A	Movies-2	Correct	1
195	U17	QtoINS	Group A	Movies-3	Correct	1
196	U17	QtoINS	Group A	Movies-4	Correct	1
197	U17	QtoINS	Group A	Movies-5	Correct	1
198	U17	QtoINS	Group A	Movies-6	Correct	1
199	U17	ChatGPT	Group A	Medal-1	Correct	1
200	U17	ChatGPT	Group A	Medal-2	Partially Correct	1
201	U17	ChatGPT	Group A	Medal-3	Incorrect	1
202	U17	ChatGPT	Group A	Medal-4	Correct	1
203	U17	ChatGPT	Group A	Medal-5	Correct	1
204	U17	ChatGPT	Group A	Medal-6	Correct	1
205	U18	QtoINS	Group B	Movies-1	Correct	1
206	U18	QtoINS	Group B	Movies-2	Correct	1
207	U18	QtoINS	Group B	Movies-3	Correct	1
208	U18	QtoINS	Group B	Movies-4	Correct	1
209	U18	QtoINS	Group B	Movies-5	Correct	1
210	U18	QtoINS	Group B	Movies-6	Correct	1
211	U18	ChatGPT	Group B	Medal-1	Partially Correct	2
212	U18	ChatGPT	Group B	Medal-2	Partially Correct	2
213	U18	ChatGPT	Group B	Medal-3	Incorrect	2
214	U18	ChatGPT	Group B	Medal-4	Partially Correct	1
215	U18	ChatGPT	Group B	Medal-5	Correct	1
216	U18	ChatGPT	Group B	Medal-6	Partially Correct	1

D.4 Task 2 Results

We present one example of data analysis with ChatGPT and another with QtoINS.

Task 2 example: U10 using ChatGPT to analyze the car dataset

U10 explored which vehicle characteristics significantly affect MPG. U10 first asked, “Show me the relationship between horsepower and MPG.” ChatGPT generated a scatterplot that revealed a clear negative correlation: as horsepower increases, MPG decreases. Next, U10 asked, “Illustrate the relationship between vehicle weight, origin, and fuel efficiency (MPG).” ChatGPT generated a scatterplot in which each dot was colored by origin, showing that European and Japanese cars clustered toward lighter weight and higher MPG, suggesting more fuel-efficient designs, while American cars were typically heavier with lower MPG. U10 then asked, “Show me the relationship between cylinders and MPG.” ChatGPT generated five boxplots for cylinders 3, 4, 5, 6, and 8. U10 inferred that vehicles with fewer cylinders (e.g., 3 or 4) tend to have higher MPG, indicating greater fuel efficiency. (However, through our verification, we found that the insight generated by ChatGPT was incorrect. In fact, cylinders (e.g., 4 or 5) tend to have higher MPG.) Finally, U10 asked, “Illustrate the relationship between vehicle weight, horsepower, and fuel efficiency (MPG).” ChatGPT generated a scatterplot with weight on the x-axis and horsepower on the y-axis, colored by MPG. The chart revealed that vehicles with lower horsepower and lighter weight generally achieve higher MPG, whereas those with higher horsepower and heavier weight tend to show lower MPG.

Task 2 example: U10 using QtoINS to analyze the insurance dataset

U10 tried to identify factors influencing insurance charges. U10 first asked “How does BMI affect insurance charges?”, QtoINS revealed a slight upward trend in a scatterplot: higher BMI is associated with higher charges. Next, U10 asked, “How does sex affect insurance charges?” and observed in a bar chart that men generally incur higher charges than women. U10 then asked, “How do BMI and sex affect insurance charges?” and found that charges increase more sharply at higher BMI levels, with males showing greater volatility. The follow-up query, “How do BMI and region affect insurance charges”, generated a scatterplot with charges on the x-axis, BMI on the y-axis, and colored by region, showing that higher BMI is associated with increased charges, particularly in the Southeast region. Finally, U10 asked, “How do BMI and smoking status affect insurance charges?” The resulting scatterplot, with BMI on the x-axis, charges on the y-axis, and colored by smoking status, showed that most data points for smokers were positioned above those for non-smokers. From this, U10 concluded that smokers face substantially higher insurance charges than non-smokers.

Table 10: Participant information for the comparative usability study, including position, education level, academic background, frequency of performing data analysis, and frequency of using LLMs.

ID	Position	Education	Major	Year of Data Analysis	Frequency of using LLM
U1	Graduate Student	Bachelor	Telecommunication Eng.	> 5 years	Occasionally
U2	Graduate Student	Bachelor	Public Health	1-2 years	Weekly
U3	Graduate Student	Bachelor	Mechanical Eng.	3-5 years	Daily
U4	Graduate Student	Master	Mechanical Eng.	3-5 years	Weekly
U5	Researcher	Doctoral	Environmental Eng.	3-5 years	Daily
U6	Graduate Student	Master	Civil Eng.	> 5 years	Daily
U7	Semiconductor R&D Engineer	Master	Materials Science	< 1 year	Daily
U8	Graduate Student	Bachelor	Mechanical Eng.	1-2 years	Daily
U9	Academic Editor	Master	Psychology	1-2 years	Daily
U10	Self-Employed Business	Bachelor	Finance	1-2 years	Daily
U11	Undergraduate Student	High school	Design	1-2 years	Daily
U12	Graduate Student	Bachelor	Mechanical Eng.	3-5 years	Weekly
U13	Graduate Student	Bachelor	Biomedical Science	> 5 years	Daily
U14	Undergraduate Student	High school	Economic	3-5 years	Daily
U15	Undergraduate Student	High school	Economic	1-2 years	Daily
U16	Product Manager	Bachelor	Materials Science	3-5 years	Daily
U17	Undergraduate Student	High school	Economic	3-5 years	Daily
U18	Undergraduate Student	High school	Finance	1-2 years	Daily

E PARTICIPANT COMMENTS FROM COMPARATIVE USABILITY STUDY

Feedback from 10 casual experts was collected after the comparative study of QtoINS and ChatGPT, focusing on the user interface, ambiguity resolution, chart generation, insight generation, and overall system comparison.

User Interface. One participant responded positively to the design of the user interface. U10 noted: “*When I used QtoINS to analyze the insurance dataset in Task 2, the dedicated UI effectively supported my analysis needs. I could ask questions while simultaneously viewing insights, and this parallel workflow significantly speeds up my data analysis.*” Most participants recognized the benefits of interactive charts rendered with Vega-Lite, noting they were substantially better than the static charts generated by ChatGPT. U6 stated: “*The charts generated by QtoINS have good interactive features, such as clicking effects on bars and data points, as well as zooming in and out of the figures.*”

Ambiguity Resolution. Participants generally appreciated the system’s ability to handle ambiguous concepts and values. U10 noted: “*I’ve used other LLM tools, such as DeepSeek for data analysis, but even when enabling the ‘deep thinking mode’, they don’t proactively clarify ambiguity for me; I have to state my needs very clearly in order to have a good chance of getting the correct answer. QtoINS is a specialized tool for analyzing datasets that meets my needs.*” Similarly, U6 highlighted the system’s handling of ambiguous values when analyzing the Medals dataset in Task 1: “*When I asks question: ‘How many total swimming medals did each of the top 5 countries win?’, QtoINS could help me identify different types of swimming, I could clearly identify that the Medals dataset contains only two categories of swimming: Swimming and Synchronized Swimming, which allowed me to easily clarify my previously ambiguous question.*” U14 commented: “*QtoINS will mention me to select data value from original question*”

Chart Generation. Participants generally appreciated the chart generation capabilities of QtoINS. U5 noted: “*The chart generated from QtoINS greatly helps me understand the trend of charges in the insurance dataset, making the results more straightforward.*” U4 noted: “*I think the interactive features of charts in QtoINS are very useful, such as hover, zoom in, and zoom out, which allow me to obtain data information more quickly and clearly. For example, when I ask, ‘What is the charge for people with a higher BMI?’ in the insurance dataset, QtoINS could leverage interactive features, such as hovering the points, to help me explore the details of insurance charges.*” U3 added: “*The chart generation from QtoINS conveys insights more intuitively without requiring line-by-line reading of the text.*” U9 commented: “*Charts are sometimes unnecessary.*”, but they highlighted the benefits of interactivity: “*When I use the car dataset and ask a question: ‘Show me the relationship between Displacement and Horsepower.’, By using the chart’s interactive featuresâĂŤfor example, hoverâĂŤI was able to quickly identify outliers and examine their values.*” U12 commented:“*The chart generated from QtoINS is better than ChatGPT*” U13 also commented:“*Compared to using ChatGPT, QtoINS offers an easier way to reshape charts.*” U15 highlighed the interactive features in QtoINS “*QtoINS provides interactive chart features, while ChatGPT only generates static charts, which don’t look as good.*”

Insight Generation. Participants expressed both appreciation and suggestions regarding insight generation. U10 appreciated the UI interactions in the Insight View and noted: “*The system provides flexibility to edit multiple insights and derive new ones through UI interactions.*” Some participants give some suggestions in Insight View. U3 provided some suggestions for this part: “*The system should allow the insights generated from the chart to be downloaded directly.*” U10 hoped that “*QtoINS currently lacks fine-grained visual linting, as well as explicit cues that directly link specific insight components to their corresponding graphical elements in the chart.*”

Comparison between ChatGPT and QtoINS. Some participants highlighted the proactive clarification function of QtoINS, which increased their awareness of ambiguities in their queries. For example, U1 noted: “*Unlike ChatGPT, which tends to assume the ambiguity in my question, QtoINS proactively clarified them, making it better suited for rigorous data analysis.*” U2 appreciated the UI design of QtoINS compared to ChatGPT: “*ChatGPT is an integrated tool that displays all the texts and charts in the same window, while in QtoINS, they were displayed in separate windows. I can find my previous analysis process more quickly in QtoINS.*” U6 valued the data summary provided by QtoINS, stating: “*I think the type of each variable showing right after loading the dataset could help me understand the dataset.*” U6 further added: “*QtoINS is super friendly to people with no background in data analysis. It provides detailed guidance to explore the dataset, with visual aids. However, it is designed to handle data analysis queries. ChatGPT can handle more complex natural language commands, [such as asking it to perform regression for data analysis,] which might not be correct, but it is capable of doing it at least, and it can learn from mistakes when provided with corrective feedback.*” They expect QtoINS to offer more sophisticated modeling functions and provide stronger statistical support. U17 commented: “*QtoINS is slower, but it understands me better than ChatGPT.*” U18 commented: “*QtoINS is more convenient for data analysis and data visualization.*”

The way to clarify ambiguity. U11 commented: “*I feel a mixed way would be helpful because the suggested algorithms save me a lot of thinking, and I probably wouldn’t come up with such complete ideas on my own.*” U12 stated: “*Checking boxes is quicker for me. Honestly, I use AI because I don’t want to type or think too much, so I prefer just clicking options.*” U17 has similar comments with U11: “*It may be beneficial to implement a hybrid model, combining automated recommendations with selectable or editable options to balance efficiency and user control.*” However, U13 expressed some concerns about using natural language to clarify ambiguity : “*If I use natural language to clarify ambiguity, it still relies on algorithms to pick the data, and no algorithm can ever be completely accurate. So even though using natural language might be more flexible, I still prefer choosing options myself. That way, I know the result is exactly what I want.*”

F LLM REQUIREMENTS AND TRADE-OFFS IN OUR SYSTEM

Table 11: Trade-off comparison between OpenAI models.

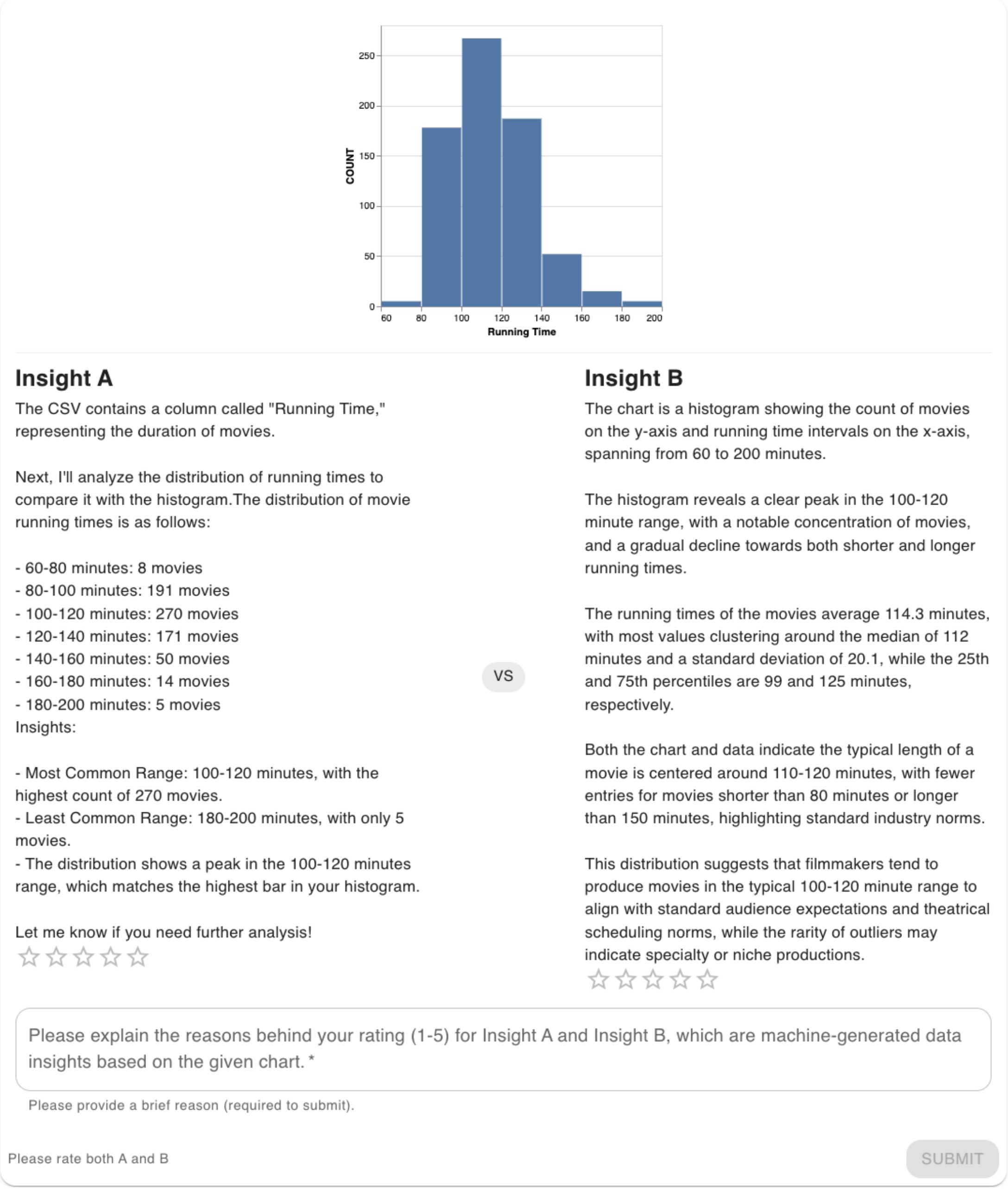
Model	Characteristics	Fine-tuning	Reasoning Tokens	Price (Per 1M tokens)	Speed	Intelligence	Max Output Tokens
GPT-4o	Fast, intelligent, flexible GPT model [46]	Available	None	Input: 2.5 Output: 10	3	3	16,384
GPT-3.5 Turbo	Legacy GPT model for cheaper chat and non-chat [46]	Available	None	Input: 0.5 Output: 1.5	1	2	4,096
GPT-4o Mini	Fast, affordable small model for focused tasks [46]	Available	None	Input: 0.15 Output: 0.60	4	2	16,384
GPT-4.5	Deprecated large model [46]	Unavailable	None	Input: 75.00 Output: 150	3	4	16,384

Table 12: The capability requirements of the model for each component.

Component	Requirement	Model Choose	Reason
Wrapped SQL Statements Generation	It requires a certain level of natural language understanding and the ability to generate Wrapped SQL Statements.	GPT-4o	The price of GPT-4o is relatively moderate, and its speed and intelligence are well balanced.
Resolve Ambiguous Concepts	We need an LLMs that can integrate the human query with the header information to recommend potential solutions for resolving ambiguous concepts.	GPT-4o	The price of GPT-4o is relatively moderate, and its speed and intelligence are well balanced.
Validate Dynamic Concepts Defined by Users On-the-fly	When the human attempts to resolve the ambiguous concept through their input, it is necessary to check whether the input conforms to grammatical rules.	GPT-4o	The price of GPT-4o is relatively moderate, and its speed and intelligence are well balanced.
Resolve Ambiguous Values	When detecting string data in an SQL statement, it is necessary to scan the strings and recommend to the human those that may contain potential ambiguity.	MPNET+ GPT-4o	Since a single column in a table may contain a large number of strings, passing all of them into an LLM would not only risk leaking sensitive information but also increase the likelihood of hallucination. Therefore, we first use the MPNet [66] to select the top N most similar strings, and then leverage the GPT-4o with the human query to provide clarification for the human.
Human Query and Target Data to Charts	Generate chart based on the human query and the targeted data obtained from QtoINS.	Fine-tuned GPT-4o	According to Table 3, we observed that the non-Fine-tuned GPT-4o performs poorly on more complex chart types, such as Grouping Line and Grouping Scatter. Therefore, we fine-tuned GPT-4o to improve its accuracy. We chose GPT-4o because, at the time we developed the system, it offered the best balance between speed and intelligence.
Five Insights Based on a Chart	Generate CoT insights, and ensure that the insights related to the chart follow common data-visualization conventions.	GPT-4o	Because we want our insights on charts to incorporate some visualization-specific knowledge, and to follow the methodology of those prior works, we added visualization-theory elements into our prompt and used a few-shot CoT approach rather than a zero-shot CoT. The reason is that zero-shot CoT cannot well accommodate visualization knowledge. Additionally, we do not use the o-series models for insight reasoning. The reason is that for those models, it is recommended to avoid chain-of-thought prompts: since they perform reasoning internally, prompting them to “think step by step” or “explain your reasoning” is unnecessary. But because we want our insights to embed some visualization knowledge, we choose not to use the o-series models.
Refine Insights	We need to generate new insights based on the human’s triggers, and we hope that the LLMs can perform web searches while maintaining a good balance between speed and quality.	GPT-4o	GPT-4o supports web search and offers a well-balanced trade-off among speed, cost, and intelligence.

G INTERFACE FOR THE PROLIFIC STUDY

We illustrate one example from the pool of 50 charts used in the study. Figure 18 shows a histogram with *running time* on the x-axis and *count of movies* on the y-axis. Five Prolific participants compared insights generated by the CoT-based approach (Insight B) and the GPT-4o API (Insight A). On average, the CoT-based approach received higher ratings ($\mu = 4.6$) than GPT-4o ($\mu = 3.8$). Four out of five participants preferred the CoT-based insight, noting that it made the chart easier to interpret.



Insight A

The CSV contains a column called "Running Time," representing the duration of movies.

Next, I'll analyze the distribution of running times to compare it with the histogram.The distribution of movie running times is as follows:

- 60-80 minutes: 8 movies
- 80-100 minutes: 191 movies
- 100-120 minutes: 270 movies
- 120-140 minutes: 171 movies
- 140-160 minutes: 50 movies
- 160-180 minutes: 14 movies
- 180-200 minutes: 5 movies

Insights:

- Most Common Range: 100-120 minutes, with the highest count of 270 movies.
- Least Common Range: 180-200 minutes, with only 5 movies.
- The distribution shows a peak in the 100-120 minutes range, which matches the highest bar in your histogram.

Let me know if you need further analysis!

☆☆☆☆☆

Insight B

The chart is a histogram showing the count of movies on the y-axis and running time intervals on the x-axis, spanning from 60 to 200 minutes.

The histogram reveals a clear peak in the 100-120 minute range, with a notable concentration of movies, and a gradual decline towards both shorter and longer running times.

The running times of the movies average 114.3 minutes, with most values clustering around the median of 112 minutes and a standard deviation of 20.1, while the 25th and 75th percentiles are 99 and 125 minutes, respectively.

Both the chart and data indicate the typical length of a movie is centered around 110-120 minutes, with fewer entries for movies shorter than 80 minutes or longer than 150 minutes, highlighting standard industry norms.

This distribution suggests that filmmakers tend to produce movies in the typical 100-120 minute range to align with standard audience expectations and theatrical scheduling norms, while the rarity of outliers may indicate specialty or niche productions.

☆☆☆☆☆

Please explain the reasons behind your rating (1-5) for Insight A and Insight B, which are machine-generated data insights based on the given chart. *

Please provide a brief reason (required to submit).

Please rate both A and B

Fig. 18: Example histogram used in the online study (Section 5.3), showing running time (x-axis) and count of movies (y-axis). Participants were asked to compare insights generated by the CoT-based approach (Insight B) and the GPT-4o API (Insight A).