Is it Right? Characterizing Errors in Large Language Model-Produced Data Visualizations

ANONYMOUS AUTHOR(S)

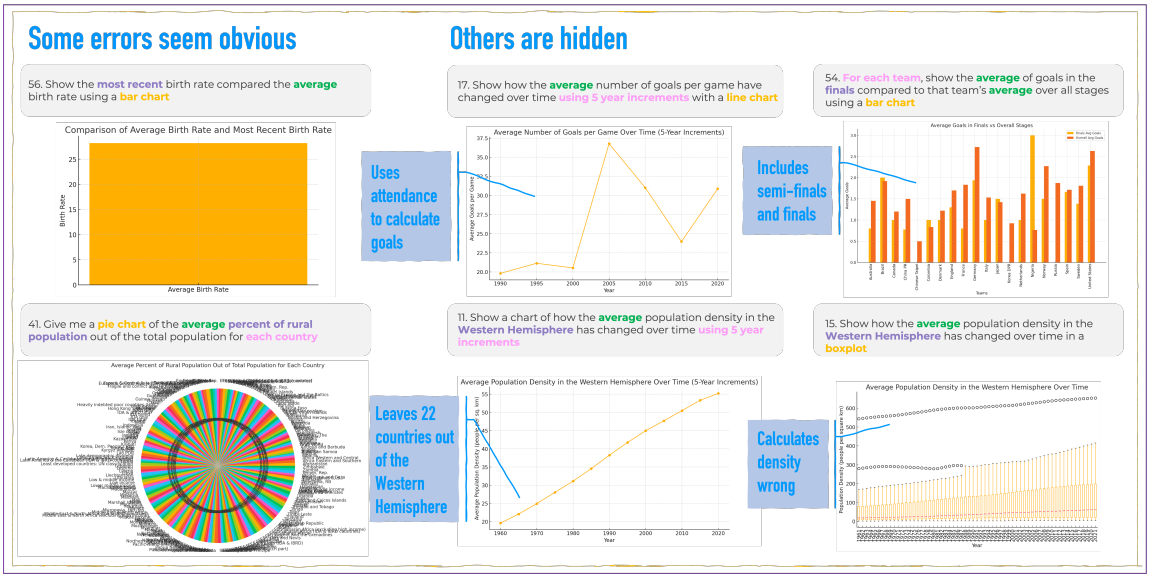

Fig. 1. Is it right? Nope, most of the time the math didn't check out. We tested 237 data visualization queries on a state-of-the-art large language model and found that while many of the produced figures and explanations looked plausible, underlying errors were frequently hidden within the code used by the model.

Large language models (LLMs) have shown potential as natural language interfaces (NLIs) for producing data visualizations. However, LLM-produced visualizations can be riddled with errors in data attribute selection, aggregation, calculation, and chart design, among others. This paper presents a characterization of the types of errors made by LLMs on data visualization (VIS) tasks. To establish our categories, we conducted a systematic study involving qualitative coding analysis of 237 query-response pairs across 2 data sets. Additionally, we provide a methodology to identify, characterize, and test for these errors. We find that a variety of errors occur throughout the data visualization pipeline and that many errors may not be easily identified without extensive work on the part of the user. Our error catalog will help developers and users of LLMs for VIS understand the applicability, reliability, and limitations of LLM-based VIS and establish the basis for developing new approaches to mitigate errors.

CCS Concepts: • **Software and its engineering** → *Correctness*; • **Theory of computation** → *Logic and verification*; • **Human-centered computing** → **Natural language interfaces**; **Empirical studies in HCI**; *Text input*; **Heuristic evaluations**; **Visualization**; **Visualization design and evaluation methods**; *Empirical studies in visualization*; **Information visualization**.

Manuscript submitted to ACM

Additional Key Words and Phrases: Machine learning, large language models, data visualization, error characterization, benchmarks, natural language interfaces.

ACM Reference Format:

Anonymous Author(s). 2025. Is it Right? Characterizing Errors in Large Language Model-Produced Data Visualizations. In . ACM, New York, NY, USA, 19 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Large language models (LLMs) are becoming a popular approach to realizing natural language interaction with data, though they come with concerns about their propensity for producing responses riddled with inaccuracies or, to borrow Hicks et al.'s language, bullshit[19]. LLM responses will inevitably include errors, but how we handle them will be just as consequential as the errors themselves. Humans developed these models; we ought to be clear-eyed about their limitations and have the patience and grace to deal with the errors they make. To build interventions that will help people recognize and respond to errors, we first need to understand and characterize what errors can occur. Therefore in this work we contribute a data study that characterizes LLM errors when they are used to create charts from data.

Through generative artificial intelligence (AI), LLMs are advancing data visualization (VIS) applications in several areas, including health, finance, and education. LLMs are now able to process data in a variety of formats uploaded directly into a model's interface and answer user queries with charts for that specific dataset. This typically involves the model generating code and providing the result of that code via a code interpreter plug-in as with GPT-4o. As noted, however, LLMs for VIS also come with concerns, including the propensity for hallucination. We lack an in-depth analysis of model error types when LLMs are used as a natural language interface (NLI) to generate a visualization. In our work, we provide a methodology to systematically identify, characterize, and test errors in LLM-based VIS to help users navigate their own VIS interactions with LLMs. Through our methodology, we conduct a data study using a popular LLM, considering both where in the VIS pipeline (i.e. the data processing steps that transform data to a visualization) an error occurred along with the query structure (see subsection 3.1). Our evaluation results in an error catalog, which characterizes the different types of errors that LLMs make on VIS tasks. This error catalog will be useful to designers and developers for testing and evaluating their models on these tasks and to users for understanding model limitations. More importantly, recognizing and categorizing these errors will help to establish a research agenda, as even when models improve, these types of errors are still likely to occur. Knowing the types of possible errors sets the stage for the development of interventions that will enable users to recognize and recover from those errors during their interaction with an NLI system.

Even for human experts, data visualization tasks can be inherently complex, setting them apart from other query types. These tasks are often complicated by a variety of factors, often linked to how well-specified a query is. Part of this complexity stems from potential query ambiguity and an extensive design space [28]. In many cases a single query may be answered in a number of ways depending on both the interpretation of query language and the design choices to be made. These, in turn, are closely bound to the nature of the dataset used. The level of measurement in the data being referenced (nominal, ordinal, interval and ratio) will limit the applicable visualizations, and at the same time, once a visualization is selected, a number of design choices must be made that impact readability and memorability[1–3, 12, 26]. These include mark selection, color selection, axes assignment, and annotation, among others.

While progress has been made in exploring the natural language to visualization (NL2VIS) space to understand how natural language utterances impact chart design [34], exploration of LLM-produced data visualization has been limited to a few studies [6, 8, 15], benchmarks [33], or specific steps of the data visualization pipeline [47]. Here, we explore the

circumstances that lead to errors (query taxonomy) and the categorization of these errors (error catalog) leveraging existing frameworks for characterizing natural language utterances [43] and visualization goals [11, 46]. Based on our resulting error catalog, we then suggest some possible mitigation approaches for future investigation. We develop our approach to be repeatable so that others can extend our results with different datasets and models.

People characterize the mistakes made by LLMs in different ways, including the common practice of referring to some errors as hallucinations [22, 35, 50]. But here, we choose to step back and, for a set of diverse visualization queries, evaluate the types of errors LLMs make through the lens of the visualization pipeline. Our research aims to answer the question, "what types of errors can occur in LLM-based NLIs for VIS and how can they be characterized?" Our aim is to provide LLM developers and users with a reference to understand LLM errors on VIS tasks and improve their senses of model applicability, reliability, and limitations. Building on NL2VIS and related NLI work, we contribute the following:

- A catalog of LLM errors organized by characteristics
- A systematic approach and repeatable methodology to allow for expansion of this catalog
- A strategy for creating query sets that span from highly specific to highly abstract and that include alternatives to core queries
- Examples of data visualization queries and dataset pairs with corresponding visualizations and categorized errors

We start with two research questions:

RQ1: What kinds of errors do LLMs make on data visualization tasks? **RQ2:** Can the errors be grouped in meaningful categories that captures useful classes of errors?

2 RELATED WORK

Here we review work on NL2VIS techniques and systems, first discussing pre-LLM approaches followed by those powered by LLMs. We also discuss benchmarks for NL2VIS systems.

2.1 Pre-LLM Natural Language Interfaces for Visualizations

Much work has been dedicated to translating from natural language to visualizations without the use of machine learning models. Several studies explore natural language utterances themselves by collecting, curating, and describing such utterances [27, 43], conducting wizard-of-oz studies [46], providing design guidelines [17, 21], and identifying characteristics of data and charts related to query language [18].

NLIs have been developed to directly translate natural language queries into visualization work. These typically use template grammars to match input queries to interpretations that can be used to generate charts. Some systems also employ statistical functions to automatically extract key facts about a dataset, visualization, or both. Cui et al. [5] explore ways to automatically generate visualizations from natural language statements. Others have created NLIs for visualizations as a way to help users explore a dataset [7, 9, 21, 38, 41]. Narechania et al.'s [36] toolkit returns an analytic specification modeled as a JSON object containing data attributes, analytic tasks, and a list of Vega-Lite specifications relevant to the input query aids visualization developers. Srinivasan et al. [44] incorporate multimodal interaction for network analysis. This type of data visualization design and exploration has also been adapted for implementation on tablets [23, 42].

Even in these early NL2VIS systems, it was well recognized that users needed support to be sure about how the system interpreted their queries and to recognize and recover from errors. Setlur et al. [40] introduced methods for

inferencing in response to underspecified utterances. DataTone introduced 'ambiguity widgets' as a mechanism to expose system interpretations of ambiguous queries and to allow error correction [14], a concept that was later extended in Eviza [38], Evizeon [21], and Orko [44]. The need for such interface supports may be even greater with LLM-based NLIs because the range of possible system actions is increased and because LLMs are known to hallucinate [22, 35, 50].

2.2 AI-based Natural Language Interfaces for Visualizations

2.2.1 Early Applications of AI to NL2VIS. AI has been studied in the space of natural language interfaces for visualizations via the exploration of user preferences in chat-style interfaces [16, 24] and through the development and testing of analytic chatbots [20, 39, 52]. Additionally, Liu et al. [31] propose an approach to automatically generate captions for visualization charts. To understand the scope of use of machine learning (ML) techniques for visualizations to achieve a better design, Wang et al. [48] survey 88 machine learning for visualization (ML4VIS) papers to identify visualization processes which can be assisted by ML and how ML techniques can be used to solve visualization problems.

2.2.2 Exploring the Use of LLMs for NL2VIS. Work on the use of LLMs for data visualization creation includes treating visualization generation as a machine translation problem. Dibia et al. [10] use a sequence-to-sequence approach to train a long short-term memory (LSTM)-based neural translation model (Data2Vis) on a corpus of Vega-Lite specifications. They also introduce language syntax validity and visualization grammar syntax validity as evaluation metrics. Tian et al. [45] decompose the generation process into a step-by-step reasoning pipeline, and allow users to check and modify the intermediate outputs of each step. Xu et al. [49] examine the capability of LLMs to perform ten low-level visual analytic tasks and demonstrate that LLMs can effectively modify existing SVG visualizations for some tasks, but perform poorly on tasks requiring mathematical operations; they also discovered that performance can vary based on factors such as the number of data points, the presence of value labels, and the chart type. Others explore and evaluate the use of LLMs for visualization generation, noting several concerns including potential shortfalls in an LLM's ability to maintain semantic intent when translating natural language questions, the risk of hallucination, the difficulties of prompt engineering, misunderstanding the data attributes, and errors in generated specifications[29, 30, 37, 49]. Lo et al. [32] also contribute work on the use of LLMs to identify misleading charts. Our work extends this line of research by formally characterizing the types of errors made by an LLM-based NL2VIS system and situating those errors in relation to steps in the visualization pipeline, while also providing contextual information about query and data structure.

2.2.3 NL2VIS Benchmarks for LLMs. Benchmarks have also been developed to test ML models on various data visualization tasks [4, 13, 33, 43, 51]. While the use of LLMs for data visualization tasks is still in its early stages, recent work examines the performance implications of AI on realistic consulting tasks, including data analysis using a spreadsheet [8]. Dell'Acqua et al. [8] found that some tasks are easily done by AI, while others remain outside of even state-of-the-art LLM capabilities. Ko et al. [25] introduce a framework for visualizations powered by LLMs that employs guided discovery and score-based paraphrasing.

Still, work to evaluate LLMs on data visualization tasks and characterize the errors LLMs make has lagged. While existing benchmarks help identify when LLMs make errors, these errors are not well-understood or categorized. To address this, we develop a systematic approach to query generation that leverages variations in query elements and goals to create query diversity. By testing a diverse set of queries for two open source datasets, we can detect both patterns in the types of errors LLMs make on data visualization tasks, and the characteristics of the prompts for each result. We provide an error catalog to help LLM developers and users better understand the nature and nuance of LLM errors in this space. Our results may contribute to future development of NL2VIS benchmarks.

3 METHODOLOGY

To build a categorization of errors and explore the circumstances that lead to them, we need a large collection of query-response pairs containing errors. To generate such a set, we need to select 1) one or more datasets to analyze with the LLM, 2) a set of data analysis queries to use as tests, and 3) one or more LLM models to test. The idea is to feed the LLM with a set of queries that represent analysis questions one may have with specific datasets. Once we obtain the results (one LLM response for each query) we need a systematic method to analyze the results, identify the errors, and build a categorization. In the following sections we provide the details and rationale of the choices we made to create the elements outlined above. We preregistered our study [1] and additional details can be found on OSF, in the supplemental material, and below.

3.1 Query Generation

To maximize the variety of errors that we would observe, we created a diverse set of queries ranging from highly specified to highly ambiguous. We established a systematic process that starts with fully specified questions across several user intent types and then removed, one-by-one, different types of query sub-clauses to create different forms of ambiguity. We complemented these queries with manually curated highly-ambiguous versions.

To ensure we had a representative and broad set of queries, we used Stephen Few's eight common relationships in charts (time series, ranking, part-to-whole, deviation, distribution, correlation, geospatial, nominal comparison) [11] and established a starter query for each of the eight types. To ensure variety in the choice of datasets we also selected two datasets which were diverse in terms of level of measurement (nominal, ordinal, interval and ratio). Our datasets were 1) https://www.kaggle.com/datasets/piterfm/football-fifa-womens-world-cup-1991-2023, FIFA Women's World Cup, 1991 - 2023, and 2) https://www.kaggle.com/datasets/nicolasgonzalezmunoz/world-bank-world-development-indicators?resource=download, World Bank Development Indicators 1960-2023.

Within each query type, many different queries can be formulated. To make the construction of queries within each query type systematic, and to create different kinds of ambiguities, we adapt Srinivasan et al.'s [43] characterization of natural language utterances which connects language elements to data operations. The query elements include: attribute, chart type, encoding, aggregation, and design.

In our use of the this categorization we found that some query features may overlap between attribute, aggregation function and design (e.g. in some cases filtering and design may serve the same purpose), so for our work we use a refined set of **query elements**: *aggregation function* (e.g. average, total, percentage, count), *measure* (value being aggregated, e.g. goals, wins), *chart type* (e.g. bar chart, pie chart, dot plot), and *grouping* (e.g. by team, continent, year). See Figure 2 for an overview of query details. Using these, we produce five versions of each of the seven starter queries - one that is fully specified, using all query elements, and four that each exclude one element. We then reviewed all queries generated by this systematic method. When the resulting query was nonsensical or when the wording was awkward compared to how a human would phrase the question, we made manual adjustments to the phrasing.

The above approach produces queries with systematic ambiguities; however, people's NL questions are often much more vague, with multiple ambiguities. To incorporate such questions in our set, we produce two ambiguous versions of each seed query that reflect a more organic natural language utterance, one that is close to the seed query in terms of at least one specified feature and query goal and one that only asks generally about the query's measure. We do this to evaluate less specified versions of the starter query, which mirror more natural human-produced queries. We start

[1] https://osf.io/ru4kt/?view_only=32001080e6c7436a9863589a0fc4750a

Goal	Full Query	Ambiguous Query	Chart options
Time Series	Show how the **average** number of goals per game in the **group stage** have changed over time **using 5 year increments** with a **[---]**	1- How have the number of goals scored changed over time? 2- Tell me something interesting about goals over time	Line chart, bar chart, dot plot, boxplot, timetable
Ranking	Show the **teams** by decreasing **average** goals for games played **in China** in a **[---]**	1- Which are the best teams? 2- How do the teams compare?	Bar chart, dot plot, boxplot
Part-to-Whole	In a **[---]** show the **percent** of total goals scored by **each country in the finals in 2022**	1- What was the percent of goals by country in 2022? 2- Tell me something about the percent of goals per team?	Bar chart, stacked bar chart, pie chart
Deviation	**For each team**, show the **average** of goals in the **finals** compared to that team's **average** over all stages using a **[---]**	1- Did teams score more than their average in their last game? 2- Tell me about recent games vs averages	Bar chart, dot plot
Distribution	Show a **[---]** of the distribution of goals scored per game by **each team** that has an **xg over 3.0**	1- What is the distribution of goals by team? 2- Tell me something about goal distribution	Bar chart, density plot, strip plot, boxplot
Correlation	Give me a **[---]** of the correlation between **average** goals and **average** attendance **for each team** where **game attendance was over 10,000**	1- Is there a relationship between attendance and goals? 2- Show me something about goals and attendance	Bar chart, scatterplot
Nominal Comparison	In a **[---]** show the **total** number of wins **by teams** for teams that have ever scored **over 4 goals in a game**	1- How many times has each team won? 2- Show me something about team wins	Dot plot, bar chart, stacked bar chart
Geospatial	Show a **[---]** of **total** wins **by continent** for teams that have an **xg under 3.0**	1- Which continents have the most wins? 2- Tell me something about games by continents	Bubble map, choropleth map, cartogram

Fig. 2. Overview of query types for the soccer dataset showing fully specified and ambiguous variations of each query type. Variations were created by removing query elements (colored) and by changing the chart type. Green = aggregation, yellow = chart type, pink = grouping, purple = filter.

by maintaining the query goal. For example "Show how the average number of goals per game have changed over time using 5 year increments" is about the patterns of average number of goals per game over time, so we use "How have the number of goals scored changed over time?" as an initial more ambiguous query. It does not specify grouping, aggregation function, or chart type. Next we make a more ambiguous version of this query by removing an additional layer of specification - "Tell me something interesting about goals over time" no longer specifies that the number of goals scored is requested.

In our study we employ 2 datasets, 8 query types, and vary between 3 and 4 query elements for a total of 237 queries. We code the corresponding 239 outputs from the LLM (the model provided two answers on two occasions leading to a total of 239 query-response results). See Table 1 for details about the variations and counts.

We chose these 237 unique queries based on both the query development process (8 types, selection of applicable chart types) and time and resource constraints. Additionally the model responded with two sets of answers to two of the 237 queries, adding two additional responses to evaluate for a total of 239 query-response pairs. With this, we have a large enough sample to test the systematic variations of each query type and query factor, while limiting us to a manageable number of query-response pairs to evaluate.

Chart Goals (from [11])	Number of Applicable Charts	Count of Query Variations
Time Series	5	23
Ranking	3	15
Part-to-Whole	3	15 (Soccer), 12 (World Bank)*
Deviation	2	11
Distribution	4	15
Correlation	2	11
Nominal Comparison	3	15
Geospatial	3	15
		120 (Soccer), 117 (World Bank)

Table 1. Queries in our study by chart goal and chart type. *Removing the filter doesn't make sense for the world bank part-to-whole query; this reduces the need for the three versions of the filter removed query - one for each of the three applicable chart types.

Exceptions to our process: We did not include aggregation in the Distribution set of queries. Including an aggregation function in this query led to a query that did not make sense given that distribution and aggregation are inherently diametrically opposed. This led to one less query in the distribution set of queries for each dataset since there was no aggregation function to remove. We used two different types of part-to-whole queries, one of which does not include a filter removal query because removing the filter rendered the query nonsensical. Because there were 3 possible chart types included for part-to-whole queries, this led to 3 fewer queries in the development dataset (117 queries) than the soccer dataset (120 queries).

3.2 Query Processing

We ran each query through a large language model (GPT-4o) [2]. We disabled model memory and ensured model-produced code was captured along with the associated model response (generated figure and text). We also cleared the model results and archive between queries to avoid memorization.

3.3 Error Coding

We performed a mixed-methods analysis combining open and focused qualitative coding of the query-result items. We began with an initial coding scheme informed by the literature and iterated on that scheme over multiple coding passes based on consensus agreement of the coders. Below, we describe the final coding scheme, which was applied to all queries in a final coding pass.

Two authors independently coded each query-result pair. Authors agreed on 80.75% of coded fields and reconciled divergent entries via pair consensus between the two coders. All data was collected directly from the model, including text and figures, code, and code results (i.e. results of the code run by the model). See supplemental material for the final code book.

Each model-generated response was first coded according to whether or not the model's response was plausible based on the query language. Each coder determined whether the LLM output looked plausible or acceptable, and whether it was in fact a plausible answer to the query (Figure 3). (Note that the output sometimes looked plausible but contained incorrect data; in this situation we would code it as Looks Plausible? yes Is Plausible? no.) Next, coders determined if the chart was correct for the query or whether there were errors regardless of plausibility (Figure 4).

[2]We note a deviation from our pre-registration, which said we would use two LLMs. This change made the coding more tractable given that there were many more errors than expected.

Open coding also led to the inclusion of codes for when a model produced text first in a response versus starting with code, along with codes for where an error occurred in intermediate steps of the model's response and when those errors were self-corrected (i.e. the model later corrected for its own error within the generated code).

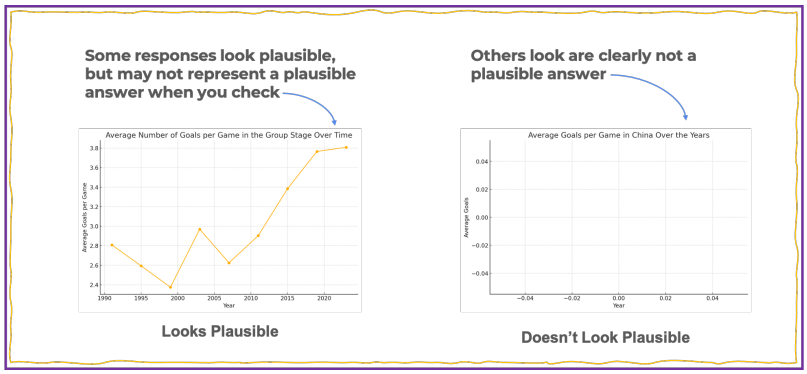

Fig. 3. Example of a plausible looking chart and a non-plausible looking chart.

Additionally, authors characterized any errors based on a modified version of a framework described by Tory et al. [46]. This included a multi-layer characterization:

(1) Where in the VIS pipeline the error occurs:
- Data attributes (e.g. Figure 5(a))
- Aggregations or calculations (e.g. Figure 5(b))
- Binning (e.g. Figure 5(c))
- Filter (e.g. Figure 5(d))
- Visualization type (e.g. Figure 5(e))

(2) Type of misrepresentation:
- Literal, where the query element is well specified, but the LLM does something different from what was asked (e.g. a bar chart in a query requesting a pie chart)

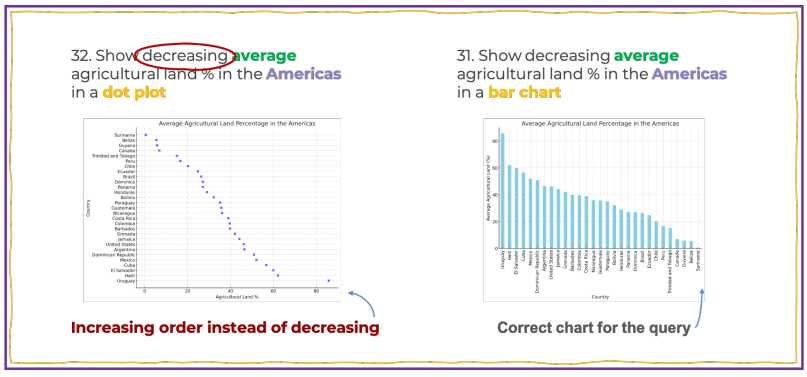

Fig. 4. Example of what is meant by correct or incorrect chart for a query.

- Poor Inference, where the query is ambiguous and the LLM makes a poor choice in inferring the missing information (e.g. failing to filter to high prices when the query requests 'expensive'; failing to include a time variable in response to "how have scores changed?")
- Whether the model self-corrected the error
- Whether the model used the information based on the error

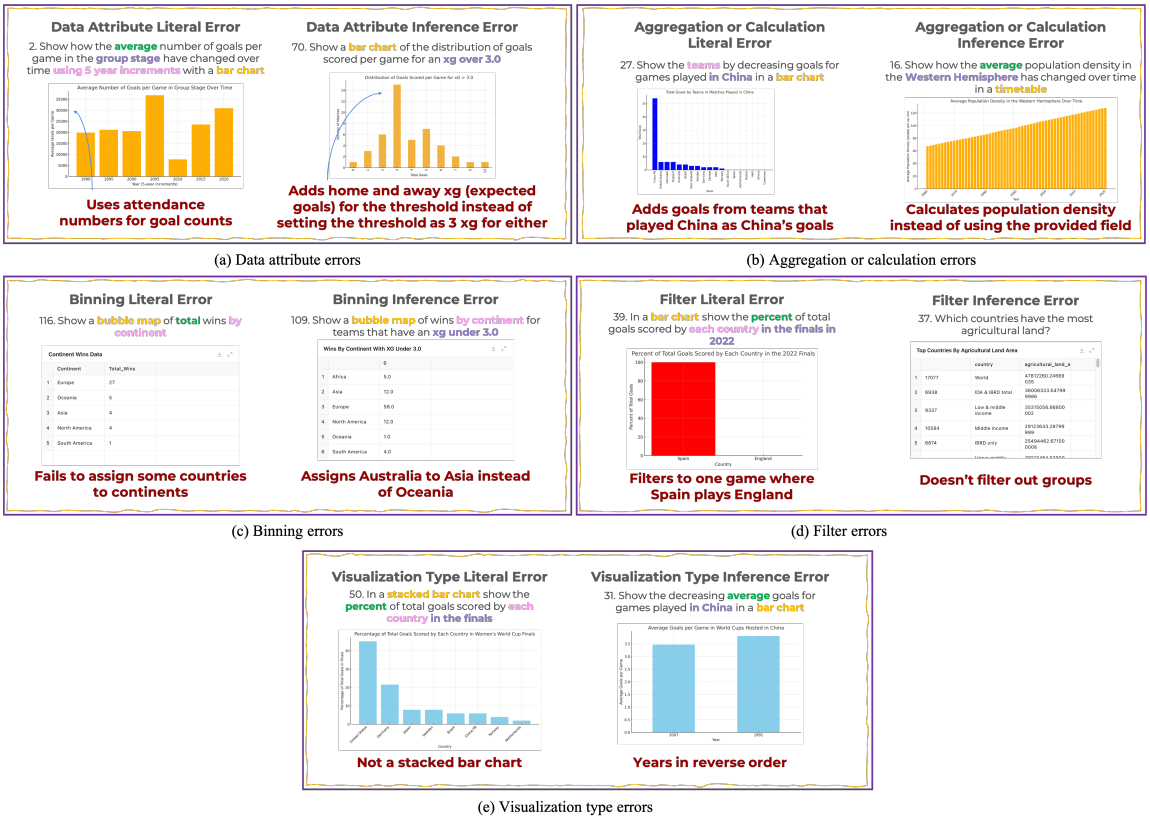

Fig. 5. Examples of error coding for portions of the visualization pipeline.

Each visualization was always accompanied by a descriptive sentence and 1-3 additional take-aways. Coders noted whether the descriptive sentence made sense for the respective visualization (Figure 6) and whether the additional sentences made sense. They also coded whether the model produced a CSV (comma separated values) file.

Authors also open-coded the errors by tagging them with additional codes representing interesting observations, such when a model's response started with text versus code (Figure 7). This included information about questionable coding choices and design choices, which were adopted into the final coding scheme.

The authors conducted a pilot using an initial set of 15 queries following this methodology for the soccer dataset. It was during this pilot that authors recognized that using *measure* as a query feature was often problematic as measure and aggregation were sometimes confounded and some form of a measure needed to be included for the other features

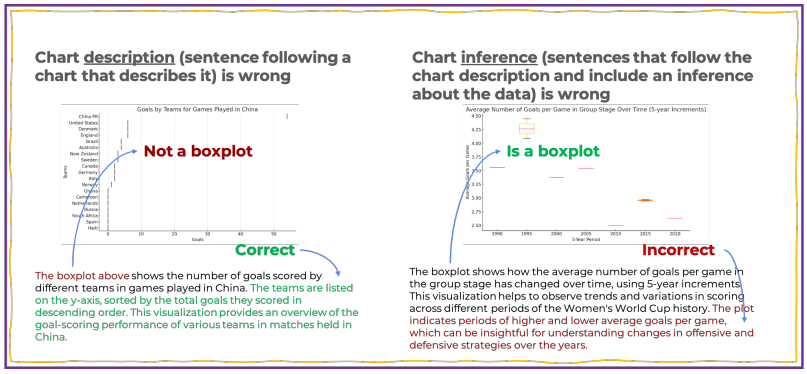

Fig. 6. Correct and incorrect chart descriptions and inferences.

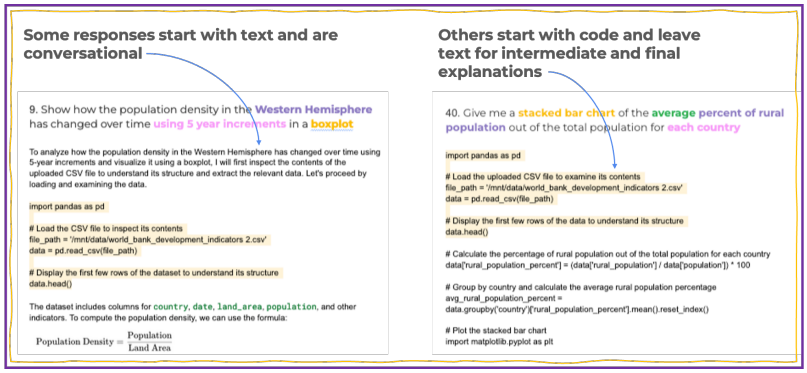

Fig. 7. Responses beginning with text vs. code.

to function. Returning to Srinivasan et al.'s work, the authors identified filter and grouping as two distinct features to test while keeping measure constant.

Based on this, the study proceeded with the coding of the following iterations of each base query:

(1) **Complete** base query
(2) Base query with **aggregation function** removed
(3) Base query with **chart type** removed
(4) Base query with **grouping** removed
(5) Base query with **filter** removed
(6) Manually generated **ambiguous** version of base query
(7) Manually generated **highly-ambiguous** version of the base query

4 RESULTS

4.1 The Plausibility Problem

As noted in Section 3, the authors used two plausibility codes; one to indicate whether a chart looked plausible as a response to its query, without checking anything, and one to indicate whether the chart was a plausible response to the

query after checking the code and values. Of the 239 model responses, 160 **(67%)** looked plausible, while only 113 **(47%)** were correct when checked. Many charts, like those in Figure 8, look correct at first glance, and were only found to have errors once authors checked the code and data. In the chart on the left the query asks, "In a bar chart show the total number of wins by team for teams that have ever scored over 4 goals in a game." The bar chart looks like a plausible answer, but after closer inspection, we realized that the generated code includes teams that never scored over 4 goals, thus failing to filter the data properly. In the chart on the right, the query asks to, "Show how the average population density in the Western Hemisphere has changed over time in a bar chart." The results looks plausible but after closer inspection we realized that the code leaves out 10 countries from the Western Hemisphere. These "hidden errors" are particularly worrisome as even observant users may not notice them. Such errors usually represented cases where the model chose the correct visual encoding, but made errors earlier in the data-to-vis pipeline, such as in calculations, aggregations, binning, or filtering steps.

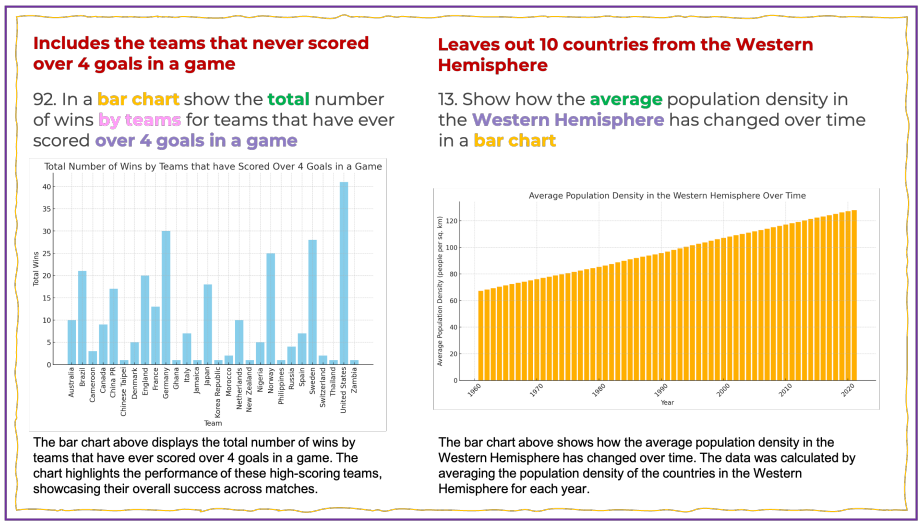

Fig. 8. Examples of responses where the charts looked plausible, but turned out to have errors.

4.2 Chart Struggles

Certain chart types also presented limitations for the model. Figure 11 shows the number of each error type by the chart type requested in a query. Pie charts often either failed entirely or were poorly constructed. None of the requested maps were generated. The model was able to generate an alternative visualization in 17 out of the 29 **(59%)** of the queries explicitly requesting maps, though 8 **(47%)** of those had errors. Some examples are presented in Figure 9.

Other charts were poorly constructed (e.g., Figure 10), suggesting that LLMs may not understand the semantic purpose of different chart types. This limitation could lead to inappropriate chart designs that might confuse rather than clarify data insights. Also, on two occasions, a chart was not produced, but when the authors ran the model's code locally, it successfully produced a chart. This indicates additional issues with the code environment.

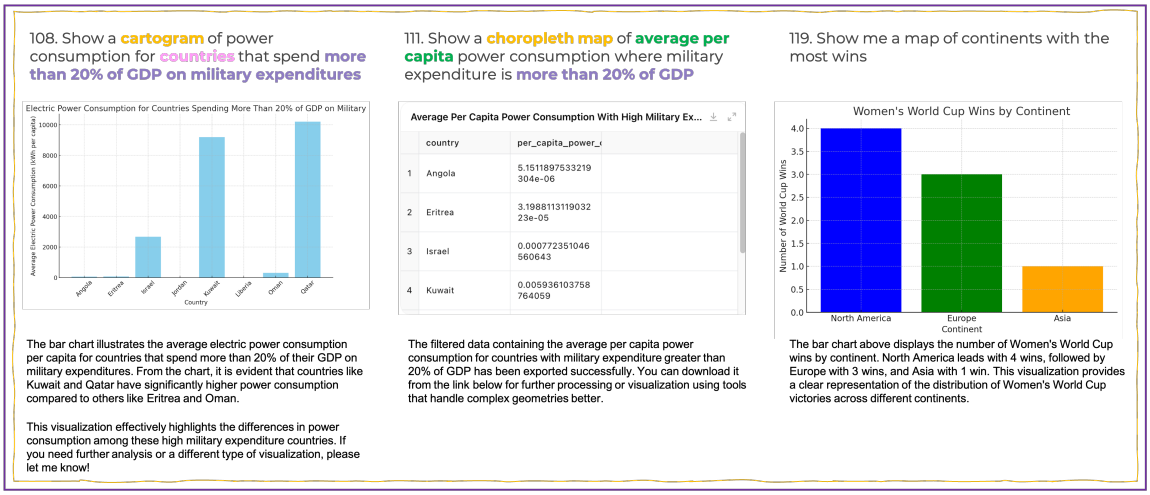

Fig. 9. Examples of failed map queries

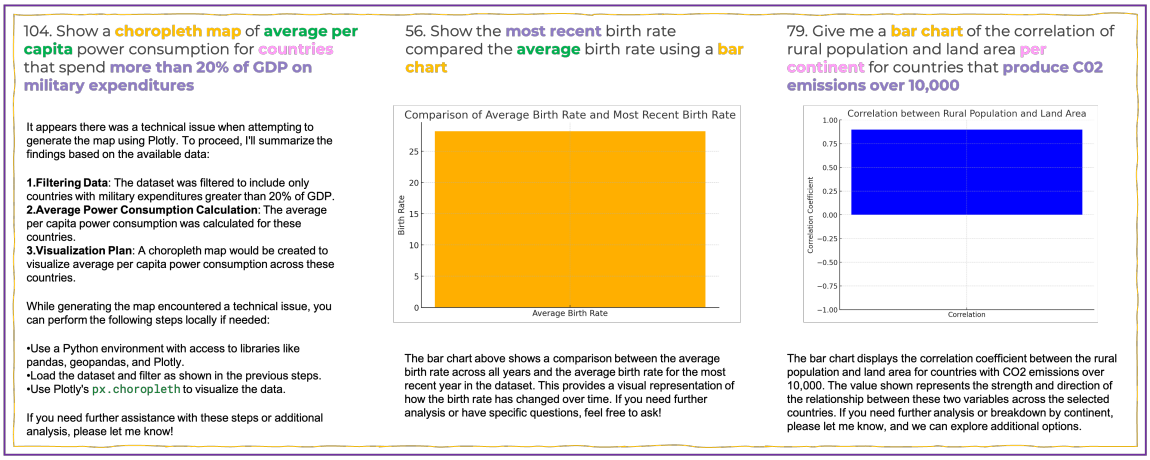

Fig. 10. Examples of responses where issues were encountered generating maps or even bar charts

4.3 Literal and Inference Errors

For each of the error types (data attribute, aggregation or calculation, binning, filter, and visualization type), the authors also coded whether the error was a literal or inference error. Examples are shown in Figure 12. Literal errors were identified in 132 **(55%)** of the 239 model responses. These were coded when explicit query requests were literally ignored in the response, such as failing to filter by group stage in response to the query, "show how the number of goals per game in the group stage have changed over time". Inference errors were identified in 173 **(72%)** of the 239 model responses. These were errors where the model correctly fulfilled the query except for some incorrect inference about the data, aggregation, binning, filter, or visualization type. See Figure 11 for a summary of literal and inference errors by chart type.

Filtering errors were common (115 or 48% of query-response pairs), especially when dealing with geographic concepts (countries, continents, etc.). Of the 16 queries requesting charts for countries in the Western Hemisphere, none correctly included all of the relevant countries from the dataset. Up to 22 countries were left out, yet each of the 16 queries varied in terms of how many and which countries were left out - between 9 and 22 were missing (these included: Antigua & Barbuda, Aruba, Bahamas, Barbados, Bermuda, British Virgin Islands, Cayman Islands, Cuba, Curacao, Dominica, Dominican Republic, Grenada, Haiti, Jamaica, Puerto Rico, St. Kitts & Nevis, St. Lucia, Saint Martin, St. Vincent and the Grenadines, Sint Maarten, Trinidad & Tobago, and Turks and Caico). When data for the Americas was requested (10 queries), between 12 and 18 countries were left out (these included: Antigua and Barbuda, Aruba, Bermuda, British Virgin Islands, Cayman Islands, Curaçao, Dominica, Greenland, Grenada, Guadeloupe, Puerto Rico, Saint Kitts and Nevis, Saint Lucia Saint Martin Saint Vincent and the Grenadines, Sint Maarten, Turks and Caicos, U.S. Virgin Islands). When Continents were requested (7), Africa was left out four times, and Australia was placed in Asia instead of Oceania.

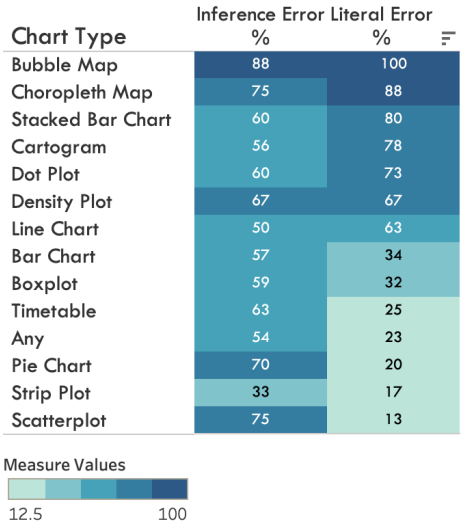

Fig. 11. Percent of charts of each type that had literal and inference errors.

4.4 Questionable Code and Code Errors

Authors identified several instances of questionable or problematic coding choices as outlined in Figure 13. This includes using game attendance to calculate goals, though a score field is provided. The most common issue identified was the use of .tail for date fields to identify the most recent data. This function returns last n rows from the object based on position. It is useful for quickly verifying data, for example, after sorting or appending rows. This is questionable because the date field may be sorted in either order, which is not accounted for in the model's code, so most recent dates may not be at the end.

4.5 Questionable Design Choices and Bad Visualizations

In many instances, authors noted problematic design choices. These include overplotting, listing years in reverse chronological order, truncating the y-axis, offsetting x-axis labels away from bars in a bar chart, among others.

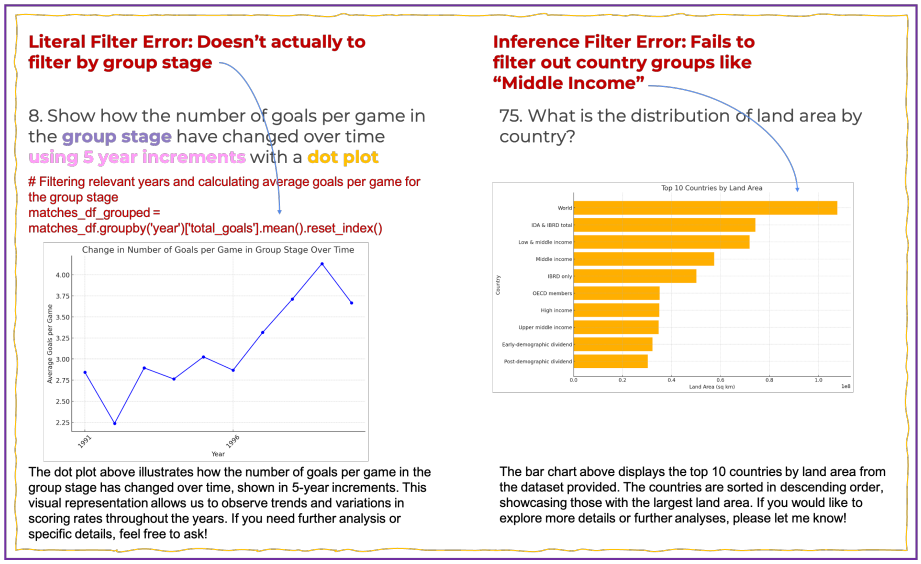

Fig. 12. Examples of literal and inference filter errors

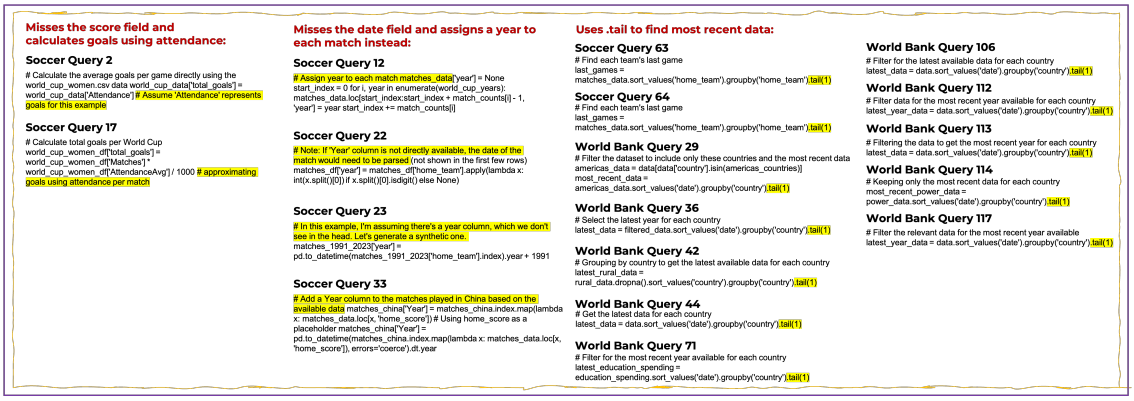

Fig. 13. Examples of questionable choices in the model's code.

Overplotting was the most common issue in the results (34), likely because of the queries requesting charts that include data for many countries. The next most common issues were truncation of the y-axis (8) and ordering in the opposite order as the query requested (8). Color selection was an issue (4) as well, as similar colors were used for different categories.

4.6 Other Patterns

Authors also identified when a model began a response with text (See Figure 7), which occurred in 87 **(36%)** of model responses. Of these responses 14 **(16%)** were correct, compared with 39 **(26%)** for non-text-first responses.

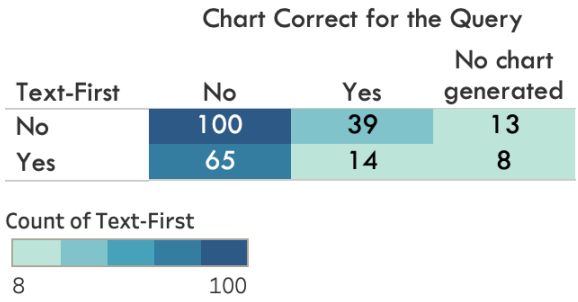

Fig. 14. A table of the number of responses where the model began with and favored text versus code, and when the respective charts were correct for the query, incorrect or were not generated.

The authors also noted when the model's description of a chart was incorrect, such as describing a regular bar chart as a stacked bar chart, and when a model's inference was incorrect, such as when the additional exposition provided after the chart claimed something about the chart or data that was incorrect. One example is when a chart is sorted in order of increasing goals, but is described as sorted in decreasing order of goals. Authors noted when a response included a CSV file produced for download, as this was never necessary to answer any of the queries. Files were produced for 51 **(21%)** of the 239 model responses.

5 DISCUSSION

LLM-based NL2VIS systems offer significant potential as supportive tools for human decision-making. They can enhance and democratize the data visualization pipeline and provide valuable assistance in domains lacking data visualization expertise due to high costs or limited resources. This capability could allow more organizations to access better data insights, ultimately improving their strategic decision-making. However, the sheer number and diversity of errors that we observed suggest that these systems may still be too immature for widespread use.

One of the most striking observations is what we term the "plausibility problem." Many LLM-generated visualizations appear correct at first glance but contain subtle (easy to miss) yet critical errors upon closer inspection. This phenomenon suggests that LLMs may be prioritizing surface-level text-coherence over deep understanding of data relationships. It poses a significant challenge for users, especially those without extensive data visualization expertise, as it may lead to misinterpretation of data and consequently poor decision-making. This finding underscores the need for robust verification mechanisms and raises important questions about trust and reliance on AI-generated content in data analysis workflows. Our analysis demonstrates that these hidden errors can occur anywhere along the visualization analysis pipeline, from data selection, to data transformation, to filtering. This observation will be critical for designing interventions that can help users recognize and recover from errors. Ambiguity widgets [14] achieved this goal for simpler rule-based NL2VIS systems, but fall short of exposing (and enabling correction) of data transformation errors like the calculations and splitting / merging of columns that the LLM frequently attempted to perform in the examples we studied.

While verifying an LLM's visualization outputs may be extraordinarily difficult for no-code users, it still remains challenging for programmers. The lack of built-in output verification represents a primary limitation of these systems, as we observed in trying to understand exactly what the LLM did in each of our cases. After the model generates code, it is executed in the background, with only the final result presented to the user, with the code proposed in an optional

view. The only method to assess the accuracy and quality of the generated code is through manual inspection, posing a significant constraint on the system's reliability. The inability to verify code raises the risk of erroneous visualizations that are difficult to analyze, leading to incorrect results and misinterpretations.

Our results also provide some insight into opportunities for LLM improvement, albeit through observations of only one current model. The distinction between literal and inference errors provides valuable insights into the nature of LLM reasoning in the visualization context (See Figure 12). Literal errors, such as producing a bar chart when explicitly asked for a pie chart, indicate basic misunderstandings of user instructions. These errors suggest that LLMs may sometimes fail to properly follow explicit instructions in complex queries. Inference errors, like misinterpreting "middle income" as a filter criterion, reveal more nuanced challenges in natural language understanding and contextual interpretation. These errors imply that LLMs may struggle with domain-specific language and concepts, highlighting the need for more specialized training in data visualization contexts. The model we studied also seemed to struggle with some chart types like pie charts and maps, pointing to gaps in the LLMs' understanding of complex visual representations. These difficulties may be traced back to the training data or their ability to interpret spatial and proportional relationships. For instance, the frequent misuse of pie charts for non-part-to-whole relationships, along with other poorly constructed charts (Figure 10) suggests that LLMs may not understand the semantic purpose of different chart types.

Our error categorization provides a framework for understanding and improving LLM-based NL2VIS systems. For researchers, it offers a foundation for the development of interactive mechanisms for verification and error correction. It also enables targeted studies into specific error types, potentially leading to more nuanced understanding and specialized model development. Developers can use it as a diagnostic tool to create more robust systems, while users and organizations can employ it as an evaluation checklist when considering adoption. The framework has potential for extension across different domains and data types, and could inform the design of more effective user interfaces. It may also serve as a starting for developing standardized evaluation metrics and benchmarks.

6 LIMITATIONS AND FUTURE WORK

While we took effort in our study to diversify the variety of errors through a systematic variation of queries, we expect that our error categorization is incomplete. Different queries, chart types, datasets, and models may yield additional error types. We also note that LLMs are being advanced rapidly, and while some of these errors may be resolved, it is highly likely that others will arise. We encourage others to extend our framework with additional findings, building on our extensible methodology.

Looking ahead, our findings suggest several promising directions for future research and development in LLM-based visualization systems. These include developing LLM output verification techniques and interfaces, fine-tuning specialized LLMs with deeper visualization knowledge, designing interfaces that effectively communicate uncertainty related to the quality of LLM outputs, and exploring hybrid methods that combine LLMs with rule-based systems. Investigating how different prompting strategies influence output quality could lead to more effective use of these models. Additionally, custom constrained languages might be useful. In the case of Code Interpreter by OpenAI, Python is the default language for code generation. While Python's versatility is advantageous, it may be excessive for specific data visualization tasks. The model often produces multiple correct responses in various forms, but this flexibility can lead to inconsistencies or errors in code execution. A potential solution is to develop a custom language or grammar, as proposed by systems like NcNet. This could limit the model to predefined tools and syntax, reducing variance and improving code quality. Finally, we encourage user studies to explore how users can detect and resolve errors in model-generated outputs and what interventions can make this process easier.

7 CONCLUSION

Our study revealed an astounding number and variety of errors made by LLMs when producing visualizations in response to natural language data queries. These errors spanned all parts of the visualization pipeline, from data selection, through data transformation, filters, and visual encoding choices. Further, many of these errors were non-obvious, exposing a plausibility problem whereby users could be easily misled into believing an (incorrect) visualization output. These results are alarming, given that LLMs are in active use for data analysis today in applications (e.g. finance) where there could be real and detrimental impacts of bad data conclusions. Clearly, there is an urgent need for improvements and guard rails in LLMs that will be used in NL2VIS systems. However, even if the LLM outputs were correct most of the time, users will still need to understand, interpret, and verify model-produced visualizations; this observation points to a critical need for research into interface support mechanisms to enable verification. Our extensible error categorization provides a foundation for such research. We invite others to extend and build on our categorization to strengthen future NL2VIS systems.

REFERENCES

[1] Michelle A. Borkin, Zoya Bylinskii, Nam Wook Kim, Constance May Bainbridge, Chelsea S. Yeh, Daniel Borkin, Hanspeter Pfister, and Aude Oliva. 2016. Beyond Memorability: Visualization Recognition and Recall. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2016), 519–528. https://doi.org/10.1109/TVCG.2015.2467732

[2] Katy Börner, Adam Maltese, Russell Nelson Balliet, and Joe Heimlich. 2016. Investigating aspects of data visualization literacy using 20 information visualizations and 273 science museum visitors. *Information Visualization* 15, 3 (2016), 198–213. https://doi.org/10.1177/1473871615594652 arXiv:https://doi.org/10.1177/1473871615594652

[3] Anne-Flore Cabouat, Tingying He, Petra Isenberg, and Tobias Isenberg. 2024. PREVis: Perceived Readability Evaluation for Visualizations. arXiv:2407.14908 [cs.HC] https://arxiv.org/abs/2407.14908

[4] Nan Chen, Yuge Zhang, Jiahang Xu, Kan Ren, and Yuqing Yang. 2024. VisEval: A Benchmark for Data Visualization in the Era of Large Language Models. arXiv:2407.00981 [cs.HC] https://arxiv.org/abs/2407.00981

[5] Weiwei Cui, Xiaoyu Zhang, Yun Wang, He Huang, Bei Chen, Lei Fang, Haidong Zhang, Jian-Guan Lou, and Dongmei Zhang. 2020. Text-to-Viz: Automatic Generation of Infographics from Proportion-Related Natural Language Statements. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (2020), 906–916. https://doi.org/10.1109/TVCG.2019.2934785

[6] Yuan Cui, Lily Ge, Yiren Ding, Lane Harrison, Fumeng Yang, and Matthew Kay. 2024. Promises and Pitfalls: Using Large Language Models to Generate Visualization Items. https://doi.org/10.31219/osf.io/au3er

[7] Raul de Araújo Lima and Simone Diniz Junqueira Barbosa. 2020. VisMaker: a Question-Oriented Visualization Recommender System for Data Exploration. arXiv:2002.06125 [cs.HC] https://arxiv.org/abs/2002.06125

[8] Fabrizio DellAcqua, McFowland Edward III, Ethan Mollick, Hila Lifshitz-Assaf, Katherine C. Kellogg, Saran Rajendran, Lisa Krayer, François Candelon, and Karim R. Lakhani. 2023. Navigating the Jagged Technological Frontier: Field Experimental Evidence of the Effects of AI on Knowledge Worker Productivity and Quality. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4573321

[9] Kedar Dhamdhere, Kevin S. McCurley, Ralfi Nahmias, Mukund Sundararajan, and Qiqi Yan. 2017. Analyza: Exploring Data with Conversation. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces* (Limassol, Cyprus) *(IUI '17)*. Association for Computing Machinery, New York, NY, USA, 493–504. https://doi.org/10.1145/3025171.3025227

[10] Victor Dibia and Çağatay Demiralp. 2019. Data2Vis: Automatic Generation of Data Visualizations Using Sequence-to-Sequence Recurrent Neural Networks. *IEEE Computer Graphics and Applications* 39, 5 (2019), 33–46. https://doi.org/10.1109/MCG.2019.2924636

[11] Stephen Few. 2004. Show me the numbers. *Analytics Pres* 2 (2004).

[12] Eric G. Freedman and Priti Shah. 2002. Toward a Model of Knowledge-Based Graph Comprehension. In *Diagrammatic Representation and Inference*, Mary Hegarty, Bernd Meyer, and N. Hari Narayanan (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 18–30.

[13] Siwei Fu, Kai Xiong, Xiaodong Ge, Siliang Tang, Wei Chen, and Yingcai Wu. 2020. Quda: Natural Language Queries for Visual Data Analytics. arXiv:2005.03257 [cs.CL] https://arxiv.org/abs/2005.03257

[14] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie G. Karahalios. 2015. DataTone: Managing Ambiguity in Natural Language Interfaces for Data Visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 489–500. https://doi.org/10.1145/2807442.2807478

[15] Sunwoo Ha, Shayan Monadjemi, and Alvitta Ottley. 2024. Guided By AI: Navigating Trust, Bias, and Data Exploration in AI-Guided Visual Analytics. arXiv:2404.14521 [cs.HC] https://arxiv.org/abs/2404.14521

[16] Marti Hearst and Melanie Tory. 2019. Would You Like A Chart With That? Incorporating Visualizations into Conversational Interfaces. In *2019 IEEE Visualization Conference (VIS)*. 1–5. https://doi.org/10.1109/VISUAL.2019.8933766

[17] Marti Hearst, Melanie Tory, and Vidya Setlur. 2019. Toward Interface Defaults for Vague Modifiers in Natural Language Interfaces for Visual Analysis. In *2019 IEEE Visualization Conference (VIS)*. 21–25. https://doi.org/10.1109/VISUAL.2019.8933569

[18] Rafael Henkin and Cagatay Turkay. 2022. Words of Estimative Correlation: Studying Verbalizations of Scatterplots. *IEEE Transactions on Visualization and Computer Graphics* 28, 4 (2022), 1967–1981. https://doi.org/10.1109/TVCG.2020.3023537

[19] Michael Townsen Hicks, James Humphries, and Joe Slater. 2024. ChatGPT is bullshit. *Ethics and Inf. Technol.* 26, 2 (jun 2024), 10 pages. https://doi.org/10.1007/s10676-024-09775-5

[20] Gan Keng Hoon, Loo Ji Yong, and Goh Kau Yang. 2019. Interfacing Chatbot with Data Retrieval and Analytics Queries for Decision Making. *Lecture Notes in Mechanical Engineering* (2019). https://api.semanticscholar.org/CorpusID:198329911

[21] Enamul Hoque, Vidya Setlur, Melanie Tory, and Isaac Dykeman. 2018. Applying Pragmatics Principles for Interaction with Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 309–318. https://doi.org/10.1109/TVCG.2017.2744684

[22] Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. 2023. Towards Mitigating LLM Hallucination via Self Reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 1827–1843. https://doi.org/10.18653/v1/2023.findings-emnlp.123

[23] Jan-Frederik Kassel and Michael Rohs. 2018. Valletto: A Multimodal Interface for Ubiquitous Visual Analytics. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI EA '18)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3170427.3188445

[24] Jan-Frederik Kassel and Michael Rohs. 2019. Talk to Me Intelligibly: Investigating An Answer Space to Match the User's Language in Visual Analysis. In *Proceedings of the 2019 on Designing Interactive Systems Conference* (San Diego, CA, USA) *(DIS '19)*. Association for Computing Machinery, New York, NY, USA, 1517–1529. https://doi.org/10.1145/3322276.3322282

[25] Hyung-Kwon Ko, Hyeon Jeon, Gwanmo Park, Dae Hyun Kim, Nam Wook Kim, Juho Kim, and Jinwook Seo. 2024. Natural Language Dataset Generation Framework for Visualizations Powered by Large Language Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 843, 22 pages. https://doi.org/10.1145/3613904.3642943

[26] Stephen M. Kosslyn. 1989. Understanding charts and graphs. *Applied Cognitive Psychology* 3, 3 (1989), 185–225. https://doi.org/10.1002/acp.2350030302 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/acp.2350030302

[27] Abhinav Kumar, Jillian Aurisano, Barbara Di Eugenio, Andrew Johnson, Alberto Gonzalez, and Jason Leigh. 2016. Towards a dialogue system that supports rich visualizations of data. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Raquel Fernandez, Wolfgang Minker, Giuseppe Carenini, Ryuichiro Higashinaka, Ron Artstein, and Alesia Gainer (Eds.). Association for Computational Linguistics, Los Angeles, 304–309. https://doi.org/10.18653/v1/W16-3639

[28] Heidi Lam, Melanie Tory, and Tamara Munzner. 2018. Bridging from Goals to Tasks with Design Study Analysis Reports. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 435–445. https://doi.org/10.1109/TVCG.2017.2744319

[29] Guozheng Li, Xinyu Wang, Gerile Aodeng, Shunyuan Zheng, Yu Zhang, Chuangxin Ou, Song Wang, and Chi Harold Liu. 2024. Visualization Generation with Large Language Models: An Evaluation. arXiv:2401.11255 [cs.HC] https://arxiv.org/abs/2401.11255

[30] Harry Li, Gabriel Appleby, and Ashley Suh. 2024. A Preliminary Roadmap for LLMs as Assistants in Exploring, Analyzing, and Visualizing Knowledge Graphs. arXiv:2404.01425 [cs.HC] https://arxiv.org/abs/2404.01425

[31] Can Liu, Liwenhan Xie, Yun Han, Datong Wei, and Xiaoru Yuan. 2020. AutoCaption: An Approach to Generate Natural Language Description from Visualization Automatically. In *2020 IEEE Pacific Visualization Symposium (PacificVis)*. 191–195. https://doi.org/10.1109/PacificVis48177.2020.1043

[32] Leo Yu-Ho Lo and Huamin Qu. 2024. How Good (Or Bad) Are LLMs at Detecting Misleading Visualizations? arXiv:2407.17291 [cs.HC] https://arxiv.org/abs/2407.17291

[33] Yuyu Luo, Jiawei Tang, and Guoliang Li. 2021. nvBench: A Large-Scale Synthesized Dataset for Cross-Domain Natural Language to Visualization Task. arXiv:2112.12926 [cs.HC] https://arxiv.org/abs/2112.12926

[34] Yuyu Luo, Nan Tang, Guoliang Li, Chengliang Chai, Wenbo Li, and Xuedi Qin. 2021. Synthesizing Natural Language to Visualization (NL2VIS) Benchmarks from NL2SQL Benchmarks. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD '21)*. Association for Computing Machinery, New York, NY, USA, 1235–1247. https://doi.org/10.1145/3448016.3457261

[35] Ariana Martino, Michael Iannelli, and Coleen Truong. 2023. Knowledge Injection to Counter Large Language Model (LLM) Hallucination. In *The Semantic Web: ESWC 2023 Satellite Events*, Catia Pesquita, Hala Skaf-Molli, Vasilis Efthymiou, Sabrina Kirrane, Axel Ngonga, Diego Collarana, Renato Cerqueira, Mehwish Alam, Cassia Trojahn, and Sven Hertling (Eds.). Springer Nature Switzerland, Cham, 182–185.

[36] Arpit Narechania, Arjun Srinivasan, and John Stasko. 2021. NL4DV: A Toolkit for Generating Analytic Specifications for Data Visualization from Natural Language Queries. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 369–379. https://doi.org/10.1109/TVCG.2020.3030378

[37] Subham Sah, Rishab Mitra, Arpit Narechania, Alex Endert, John Stasko, and Wenwen Dou. 2024. Generating Analytic Specifications for Data Visualization from Natural Language Queries using Large Language Models. arXiv:2408.13391 [cs.HC] https://arxiv.org/abs/2408.13391

[38] Vidya Setlur, Sarah E. Battersby, Melanie Tory, Rich Gossweiler, and Angel X. Chang. 2016. Eviza: A Natural Language Interface for Visual Analysis. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing

Machinery, New York, NY, USA, 365–377. https://doi.org/10.1145/2984511.2984588

[39] Vidya Setlur and Melanie Tory. 2022. How do you converse with an analytical chatbot? revisiting gricean maxims for designing analytical conversational behavior. In *Proceedings of the 2022 CHI conference on human factors in computing systems*. 1–17.

[40] Vidya Setlur, Melanie Tory, and Alex Djalali. 2019. Inferencing underspecified natural language utterances in visual analysis. In *Proceedings of the 24th international conference on intelligent user interfaces*. 40–51.

[41] Arjun Srinivasan, Steven M. Drucker, Alex Endert, and John Stasko. 2019. Augmenting Visualizations with Interactive Data Facts to Facilitate Interpretation and Communication. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 672–681. https://doi.org/10.1109/TVCG.2018.2865145

[42] Arjun Srinivasan, Bongshin Lee, Nathalie Henry Riche, Steven M. Drucker, and Ken Hinckley. 2020. InChorus: Designing Consistent Multimodal Interactions for Data Visualization on Tablet Devices. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376782

[43] Arjun Srinivasan, Nikhila Nyapathy, Bongshin Lee, Steven M. Drucker, and John Stasko. 2021. Collecting and Characterizing Natural Language Utterances for Specifying Data Visualizations. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 464, 10 pages. https://doi.org/10.1145/3411764.3445400

[44] Arjun Srinivasan and John Stasko. 2018. Orko: Facilitating Multimodal Interaction for Visual Exploration and Analysis of Networks. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 511–521. https://doi.org/10.1109/TVCG.2017.2745219

[45] Yuan Tian, Weiwei Cui, Dazhen Deng, Xinjing Yi, Yurun Yang, Haidong Zhang, and Yingcai Wu. 2024. ChartGPT: Leveraging LLMs to Generate Charts from Abstract Natural Language. *IEEE Transactions on Visualization and Computer Graphics* (2024), 1–15. https://doi.org/10.1109/TVCG.2024.3368621

[46] Melanie Tory and Vidya Setlur. 2019. Do What I Mean, Not What I Say! Design Considerations for Supporting Intent and Context in Analytical Conversation. In *2019 IEEE Conference on Visual Analytics Science and Technology (VAST)*. 93–103. https://doi.org/10.1109/VAST47406.2019.8986918

[47] Qianwen Wang, Zhutian Chen, Yong Wang, and Huamin Qu. 2022. A Survey on ML4VIS: Applying Machine Learning Advances to Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 28, 12 (dec 2022), 5134–5153. https://doi.org/10.1109/TVCG.2021.3106142

[48] Qianwen Wang, Chen Zhu-Tian, Yong Wang, and Huamin Qu. 2022. A Survey on ML4VIS: Applying Machine Learning Advances to Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 28, 12 (2022), 5134–5153. https://doi.org/10.1109/TVCG.2021.3106142

[49] Zhongzheng Xu and Emily Wall. 2024. Exploring the Capability of LLMs in Performing Low-Level Visual Analytic Tasks on SVG Data Visualizations. arXiv:2404.19097 [cs.HC] https://arxiv.org/abs/2404.19097

[50] Jia-Yu Yao, Kun-Peng Ning, Zhen-Hui Liu, Mu-Nan Ning, Yu-Yang Liu, and Li Yuan. 2024. LLM Lies: Hallucinations are not Bugs, but Features as Adversarial Examples. arXiv:2310.01469 [cs.CL] https://arxiv.org/abs/2310.01469

[51] Yuge Zhang, Qiyang Jiang, Xingyu Han, Nan Chen, Yuqing Yang, and Kan Ren. 2024. Benchmarking Data Science Agents. arXiv:2402.17168 [cs.AI] https://arxiv.org/abs/2402.17168

[52] Qiyu Zhi and Ronald Metoyer. 2020. GameBot: A Visualization-augmented Chatbot for Sports Game. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–7. https://doi.org/10.1145/3334480.3382794

Received 12 September 2024; revised ; accepted